MAKE | BUILD | HACK | CREATE

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.CC | June 2020 | Issue #31

CIRCUIT PYTHON GRAPHICS

MATHS FOR MAKERS

DIY SMART HOME

Build your own Internet of Things

June 2020
Issue #31 £6

Homemade DAB radio

3D modelling with Tinkercad

Raspberry Pi High Quality Camera

Liz Clark

Maker, musician, author, and sharer



PVC PIPES SENSORS COMMODORE 64 PRINT PETG

1.8 MILLION+ PRODUCTS IN STOCK | 9 MILLION+ PRODUCTS ONLINE

Semiconductors
Passives
Electromechanical
Power
Circuit Protection
Automation
Connectors
Interconnect
Hyperfast
IoT
Switches
RFID
TMR Magnetic Sensors
RF Directional Couplers
Bipolar Digital Latching Sensor
Logic
Digital Omnipolar
Crystals
Augmented Reality
Earth-Friendly Display
Embedded Cellular
IO-Link
Solenoids
Proximity Sensor
Capacitive Touch
Embedded Computers
Thermocouple Interface
PIR Sensor
SPI Interface
Linear
Ultra Low-Power
Narrowband
Mesh-Networked
Virtual Reality
Keyfob
Isolators
MCUs
RF Evaluation
Dev Boards
RF Antennas
Axis Tilt
Zettabyte Era
I2C
Robotic Process Automation
Microservice Architecture
Rezence Compatible
XCVR
Immersive Experience
Artificial Intelligence
Internet of Things
Na-TECC
3D Alteration
Quantum Computing
Shunt Sense
Touchless
Smart Home Technology
5G Mobile
Energy Harvesting
MotorData Acquisition

YOU INNOVATE!

Infrared Thermopile
Triboelectric
Magnetic Position
Gesture Control
Interconnect
Rectenna
Connected Cloud
Decoupled Network
Hybrid Envelope-Tracking Signal
Re-Entrant Leveraged Design
Embedded Logic
TEGs
Passives
Logic Eco-System
Third-Order Sensor
Clock/Timing
Memory
Filters
SoC
Thermal Management
Class-G Amplifier
Decimated Power-Efficiency
Microwave
Bluetooth
Remote Control
FPGA
DDS
Batteries
Betavoltaics
MiWi Transceiver
Nanogenerators
AMR
Recycling Radiowaves
Ask Receiver
Transformers
Solar
Sensor
2-Way Remote
Simplex Transmission
ADC
Potentiometers
Interface
NFC
Frequency Synthesizers
Oscillators
Low Energy
PMIC
Relays
WPC-Certified
Smart Devices
Capacitors
Electromechanical
Optoisolators
ZigBee
Semiconductors
EMI
Tools
Hardware
Cable

We Make It Easy.

FREE SHIPPING ON ORDERS OVER £33 OR $50 USD*

Digi-Key® ELECTRONICS

0800 587 0991
DIGIKEY.CO.UK

1,000+ INDUSTRY-LEADING SUPPLIERS | 100% FRANCHISED DISTRIBUTOR

*A shipping charge of £12.00 will be billed on all orders of less than £33.00. A shipping charge of $18.00 USD will be billed on all orders of less than $50.00 USD. All orders are shipped via UPS, Federal Express, or DHL for delivery within 1-3 days (dependent on final destination). No handling fees. All prices are in British pound sterling or United States dollar. Digi-Key is a franchised distributor for all supplier partners. New products added daily. Digi-Key and Digi-Key Electronics are registered trademarks of Digi-Key Electronics in the U.S. and other countries. © 2020 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA

ECIA MEMBER
Supporting The Authorized Channel



Welcome to HackSpace magazine

If you're reading the paper version of this magazine, you might notice that it feels a bit lighter in your hand. There are two reasons for this. The first is that we're using slightly lighter paper than in previous issues, and the second is that we're now 116 pages per month, rather than 132.

It's no secret that the print media business is tough and has been getting tougher for over a decade, but I don't think even the most pessimistic forecasters predicted the challenges of the last few months. At 116 pages, we can ensure that we're building a sustainable future for our magazine, so you'll continue to get the best makes, hacks, tutorials, and features every month.

This month, we're focusing our attention on smart homes, which is one area makers can have a real impact on the quality of their personal environment. Whether it's monitoring the temperature, controlling the lighting, or ensuring you've got a perfectly brewed cup of coffee ready for when you wake up, a smart home can make your life better. Turn to page 36 to find out how.

BEN EVERARD
Editor ben.everard@raspberrypi.org

PAGE 48
SUBSCRIBE TODAY



Got a comment, question, or thought about HackSpace magazine?

get in touch at **hsmag.cc/hello**

GET IN TOUCH

hackspace@raspberrypi.org
hackspacemag
hackspacemag

ONLINE

hsmag.cc

Available on the App Store

GET IT ON Google Play

EDITORIAL

Editor
Ben Everard
ben.everard@raspberrypi.org

Features Editor
Andrew Gregory
andrew.gregory@raspberrypi.org

Sub-Editors
David Higgs, Nicola King

DESIGN

Critical Media
criticalmedia.co.uk

Head of Design
Lee Allen

Designers
Sam Ribbits, Harriet Knight, Ty Logan

Photography
Brian O'Halloran

CONTRIBUTORS

Lucy Rogers, Drew Fustini, Jo Hinchliffe, PJ Evans, Christine Thompson, Mayank Sharma, Andrew Lewis, Poppy Mosbacher, Gareth Branwyn, Les Pounder, Marc de Vinck

PUBLISHING

Publishing Director
Russell Barnes
russell@raspberrypi.org

Advertising
Charlie Milligan
charlotte.milligan@raspberrypi.org

DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6, The Enterprise Centre,
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE

To subscribe
01293 312189
hsmag.cc/subscribe

Subscription queries
hackspace@subscriptionhelpline.co.uk

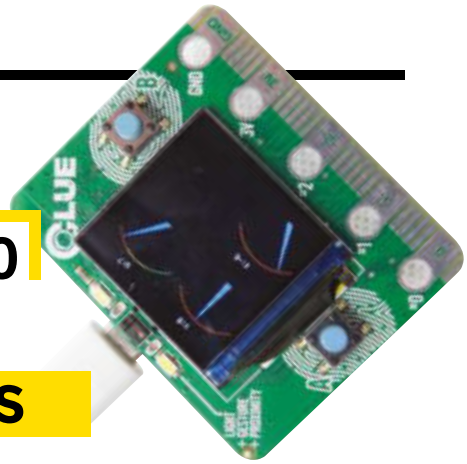
recycle

CC BY NC SA

This magazine is printed on paper sourced from sustainable forests. The printer operates an environmental management system which has been assessed as conforming to ISO 14001.

HackSpace magazine is published by Raspberry Pi (Trading) Ltd., Maurice Wilkes Building, St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0). ISSN: 2515-5148.

Contents



70

06 SPARK

06 Top Projects
Great things made by great people

18 Objet 3d'art
Perfection in hot plastic

22 Digital Making at Home
Young people: log on and learn!

24 Meet the Maker
Learn to fail = learn to make better

30 Column
Get out into the wild green yonder

32 Letters
Arduino, feedback, and failure

35 LENS

36 DIY Smart Home
Build a bespoke, secure IoT at home

50 How I Made: Seven-segment clock
Salvaged electronics, mahogany, Arduino, and ESP

54 In the workshop: Car stereo hacking
Take old gear apart to give it new life

56 Interview: Kyle Wiens
The founder of iFixit on repair, tinkering, and ownership

64 Improviser's Toolbox: PVC pipes
Make unique things from standard plumbing parts

Cover Feature



36

Tutorial

Vinyl cutter



80 Make pop-up greetings cards with CNC for paper



50



54



100



06

Can I Hack it?

Commodore 64



106 Get under the skin of one of the greatest ever computers

Interview

Kyle Wiens



56 We're fighting a war against the universe. Are you in?



110

69 FORGE

70 **SoM** CircuitPython
Draw analogue-style gauges with real data

72 **SoM** Bench blocks
Grip work more securely with homemade tools

76 **SoM** 3D printing
Everything you need to know about using PETG

78 **Tutorial** Maker maths
The quick calculations you need to make better

80 **Tutorial** Pop-up card making
Use a vinyl cutter to build paper designs

86 **Tutorial** Pallet clock
Create a timepiece with reclaimed wood

90 **Tutorial** Tinkercad
Design and print a medieval-themed planter

94 **Tutorial** CMOS music
Build a low-voltage pluggable synth

99 FIELD TEST

100 **Best of Breed**
Build your own fun with maker kits

106 **Can I Hack It?**
A beige, nicotine-stained Commodore 64

108 **Review** Pimoroni Enviro+ Wing
Add environmental testing to your Feather setup

110 **Review** Raspberry Pi HQ Camera
A huge upgrade on your old Raspberry Pi Camera Module

112 **Review** Ultimate Box Maker
3D-print enclosures for your next project

113 **Book Review** Micro:bit for Mad Scientists
Projects and experiments for techie kids

Some of the tools and techniques shown in HackSpace Magazine are dangerous unless used with skill, experience and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. HackSpace Magazine is intended for an adult audience and some projects may be dangerous for children. Raspberry Pi (Trading) Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in HackSpace Magazine. Laws and regulations covering many of the topics in HackSpace Magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in HackSpace Magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

Alien face mask

By Lady Frankenstein

hsmag.cc/SNXrQ5

No trip to an alien planet is complete without snazzy face attire these days, and we've found the perfect mask for the next time we venture out into the seething mass of humanity that is The Outside. Made by German artist Lady Frankenstein, this is just the thing to make sure the mob stays two metres away from you.

It's made of modelling clay, attaches snugly around the face and neck, and took around 20 hours of work – not surprising when you see how detailed it is.



Right
A perfect organism… Its structural perfection is matched only by its hostility



Hardware hornet

By Ross the Random

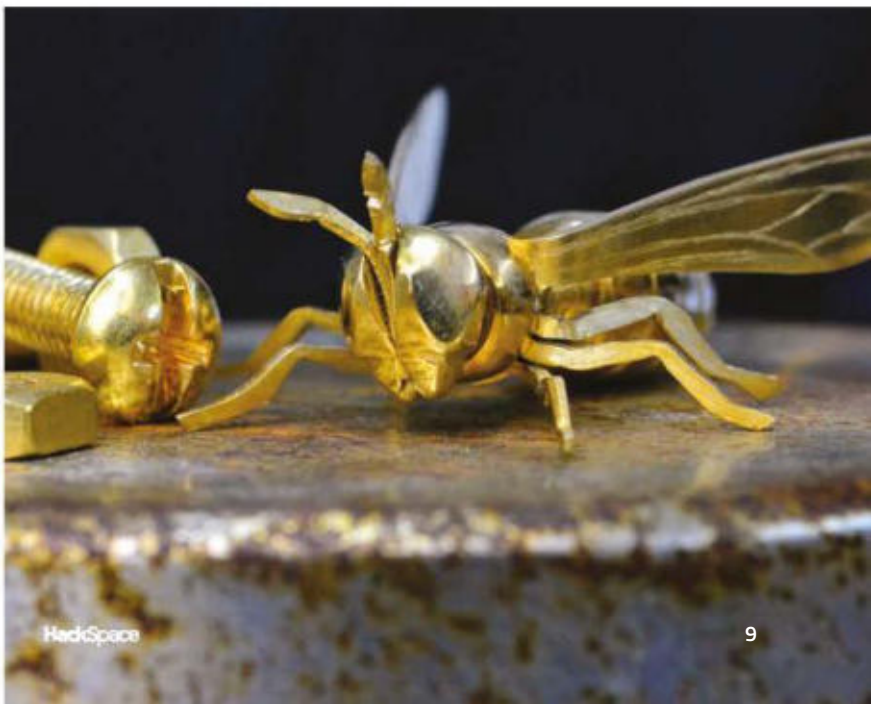
hsmag.cc/EzOoBU

So often with making, it's not the final object that's the goal; it's the process itself. This hornet is a beautiful object, but the accompanying video by Ross the Random gives it provenance, and takes it to the next level of craftsmanship.

Every part of this sculpture is made from nuts and bolts, ground, filed, hammered, cut, and polished to perfection. An insect's body is made up of sections, which correspond to differently sized nuts on a central bolt; you can see clearly the joins between the head, thorax, and abdomen.



Right
We challenge Ross to make a millipede in this style!


HackSpace

Digital camera

By Richard Hayler

hsmag.cc/kp43mw

This is a perfect example of a minimum viable product: a Raspberry Pi High Quality Camera module, an Adafruit 2.8-inch touchscreen, a Raspberry Pi, and an official Raspberry Pi case, plus a big red button hot-glued to the top. It's crude and a bit hacky, but it works perfectly.



Right
Just add a battery pack, and you're ready for the Serengeti


V1.0 2018
TAMRON
1/1.2 8mm F/2.4 C
0.15 0.3 ∞ m

Profan-O-Matic

By **Matthew** etsy.com/uk/shop/MWWorksUK

If you're the sort of person who looks in the dictionary to find the definitions of rude words, you might like this project from Suffolk-based furniture maker (check out his stuff here: mwworks.uk) and electronics tinkerer Matthew. It stands on the shoulders of Scott Bezek's split-flap project, and adds rude words, swaps out an Arduino for an ESP device, and makes the construction far more refined, accurate, and repeatable. Flippin' brilliant!

The user interface is as simple as can be: it's just four buttons. Press the left-hand button, and the device will tell the time, the green button displays a random word of general vocabulary, the yellow button picks a word from a more risqué list, and the red button displays a random swear-word. Some say swearing is a sign of a poor vocabulary, but when it makes such a glorious clicky-clack noise as this does, who cares? ◻



Right
The Profan-O-Matic's enclosure is built from 6mm oak-veneered MDF


PROFAN-O-MATIC 455
407

Freeform Sonos remote station

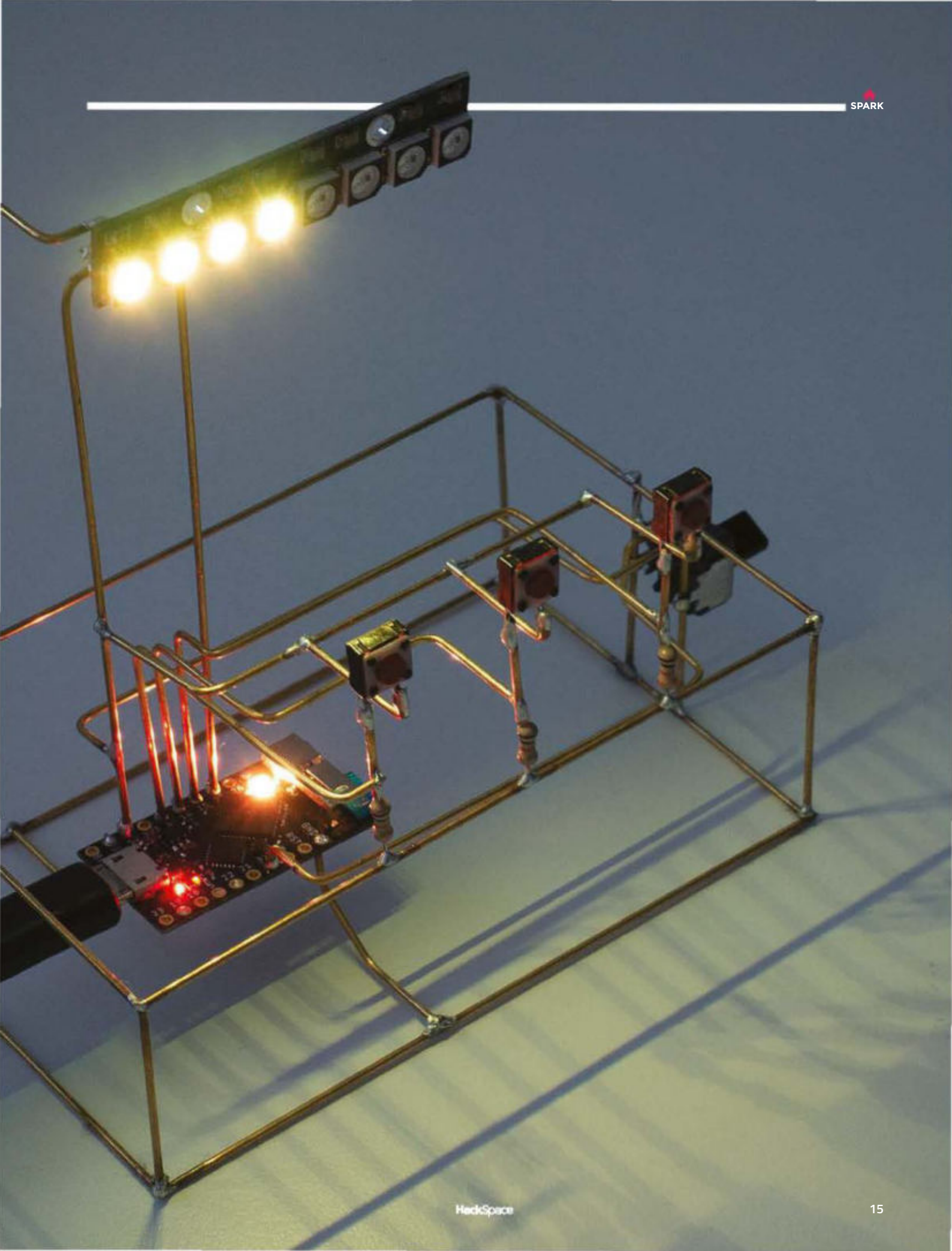
By Tom Janssens tomjanssens.de

You can usually control smart devices via an app on your phone, but there's something nice about proper tactile buttons and switches that the sterile coldness of a touchscreen lacks. That's what we think, anyway.

Tom Janssens agrees – he's built this controller for his Sonos smart speakers, inspired by the freeform soldering creations of Mohit Bhoite and Jiří Praus. It's built around a TinyPICO ESP32 dev board, and controls volume, play/pause, next track, and previous track.



Right
Tom's device connects to his speakers via the UPnP protocol



REGULAR



Time Measure

By Alex Fiel and Anna Lynton

hsmag.cc/nY2qul

With a real-time clock module and some sort of microcomputer, you can turn anything that moves into a clock, even if it was meant to measure distance, rather than time. Alex Fiel and Anna Lynton have, between them, taken the guts of a tape-measure and added an Arduino Nano, an Adafruit RTC, and a motor to make a clock that displays the time in inches.

There's a little more to it than that; because the tape has to go in and out, they need to add an H-bridge to the stepper motor, which also needed a boost/buck converter. To contain the extra electronics, they needed to make their own case, which is a 3D-printed version of a standard tape-measure housing, made big enough to fit the moving parts and the Arduino.





Left
The Time Measure runs off a hacked 12V power supply

Objet 3d'art

3D-printed artwork to bring more beauty into your life



We've had the chance to play with a Raspberry Pi High Quality Camera this month, and jolly good fun it is too. If that's piqued your interest in photography, here's the SCURA 3D-printed pinhole camera, made by Dora Goodman.

The most primitive pinhole cameras use a flat surface on to which they project and then capture an image. Goodman's camera uses a curved surface so the edges of the image aren't distorted. For such an old technology, the images created on this pinhole camera are amazingly good, and show just what you can achieve with a 3D printer and massive amounts of knowledge and hard work.

The SCURA is open source, so you can download the STL files and print your own, or, if you don't trust the accuracy of your printer, you can buy a kit from **doragoodman.com**.

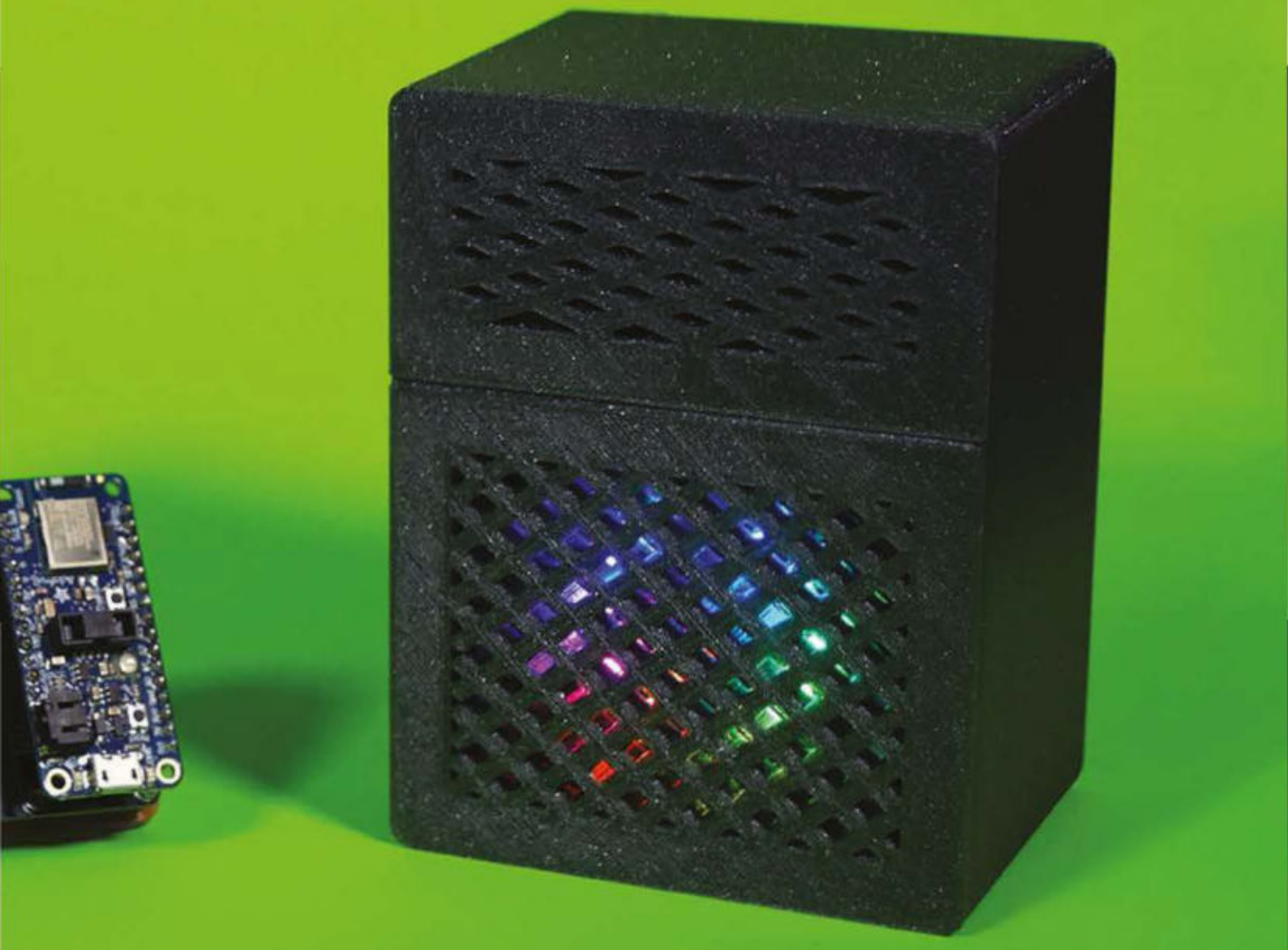


This right here is a BLE (Bluetooth Low Energy) synthesiser connecting to a speaker over the magic of BLE. According to its maker, Liz Clark (who you can hear more from on page 24), this is "the cutest keyboard ever to exist". The keys are shorter and squishier than normal, which gives the whole project a nice accessible feel.

The electronics are pretty simple too: powering the circuit on the keyboard there's a Feather nRF52840, which communicates over BLE with an Adafruit Circuit Playground Express, plus an Adafruit STEMMA Speaker (it's got an amplifier built-in) in the speaker. Aside from a battery, perfboard, jumper leads, and a handful of momentary switches, that's the entire bill of materials.

Want to make one yourself? Check out the build video at **hsmag.cc/G5zCsy**.





FEATURE

Digital Making at Home

Learning online across the world

When we talk to makers, we often hear that they don't know how to code; they just copy and paste bits and bobs from the internet until their device does what they want it to do. If it works, it works, and we're not going to judge. But if you've ever seen how intuitively some people can code, and want to pass that skill on to the young people in your life, we have to point you in the direction of Digital Making at Home, a recent and ongoing initiative from the Raspberry Pi Foundation.

Digital Making at Home is a weekly content series for young people who are creating at home with code. No prior knowledge is required – there are three difficulty levels, from basic introductory coding in Scratch, to working with external libraries in Python. Lessons are self-contained and don't need a great deal of parental input, so if you're not confident with lists, parentheses, objects, and statements, it doesn't matter in the slightest.

Each week there's a theme to spark learners' creativity: we've had games, mindfulness, jokes, celebrate our planet, and the perennial favourite, music. Each lesson is structured as you would a hardware project, with a goal in mind, rather than a look into coding for coding's sake. There's no Hello World here.

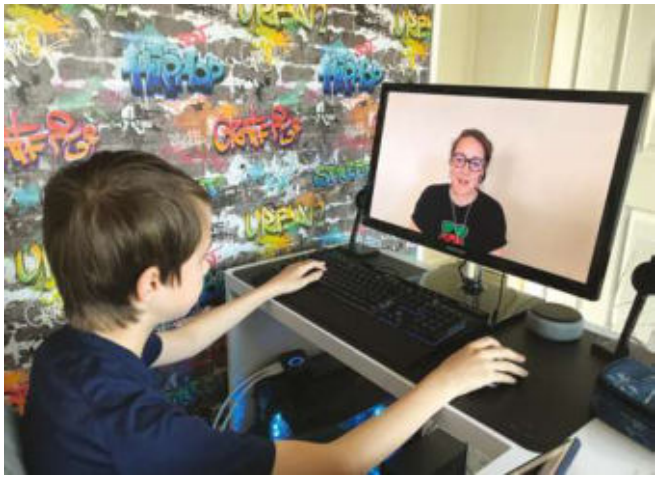
Thousands of people have signed up for the programme, with kids from all over the world sharing the things they're making.

Young learners can use the skills they've gained to create their own versions of projects – while they're learning programming, they're not just gaining technical skills, they're gaining an ability to express themselves creatively, and that's why the sharing element is so important. It's enlightening to see children from opposite sides of the world see the same problems.

For more information on Digital Making at Home, check out the blog: hsmag.cc/QLRTGp.

To sign up for Digital Making at Home, visit: hsmag.cc/pbJRrN. □





Above
There are more hours of online instruction on the internet than you could ever hope to watch – we're planning to do a couple of free courses from Harvard University, just because they're there

Left
Raspberry Pi isn't just a computer: there's a whole bunch of people behind the scenes offering learning materials, free online tutorials, support for teachers, lesson plans…



Above
Kids are prompted to consider what information they should be sharing on the internet before they post their makes online

REGULAR

Meet The Maker: Liz Clark



She makes videos about things that she makes...

As her online alter ego Blitz City DIY, **Liz Clark makes videos, music, electronics, 3D prints, tinkers with electronics, and writes code.** Even better, she shares it, free of mythology or mystery, so anyone can follow along and learn for themselves. We spoke from a socially responsible distance of 3100 miles, to find out what it's like to show the world as you teach yourself.

"I feel like there's almost two schools of makers: there's the ones that are engineers in their daily life, and they do these fun projects on the side, and there's also people like me who see these things and think 'OK, maybe I can do that?' You follow the tutorial, and you can work yourself up to trying it for yourself… it takes some time, and it takes some failure, so if I can make that a more acceptable thing to be, then I'm happy to do it.

> "I do a little bit of everything on my YouTube channel: electronics, Arduino, CircuitPython, also some Raspberry Pi and Linux things"

"I like to talk about failure because I am self-taught in all this stuff. I know from my perspective sometimes it can be intimidating to jump into these projects, and especially when you see all these beautiful polished things that people put up, and you're like, 'Oh gosh, they just know how to do this; they have all this knowledge, and it's not an issue.' If I can make people feel a bit more comfortable in starting, and realise that they're going to fail, but that's OK, and they can learn from that – that's why I started putting it out there.

"I even started doing skits to laugh at myself a bit before someone else has the opportunity to say something rude.

MAKE ALL THE THINGS

"I do a little bit of everything on my YouTube channel: electronics, Arduino, CircuitPython, also some Raspberry Pi and Linux things, a little bit of 3D printing, and a lot of music things. I've been a musician since I was a kid, and started to get back into that. I centre a lot of my electronics projects around music.

"I majored in music technology in college, and around the middle of my education, I started to feel like I wasn't learning enough to get a job. I got away from music for a couple of years, and when I started my YouTube channel, that's when I got back into music.

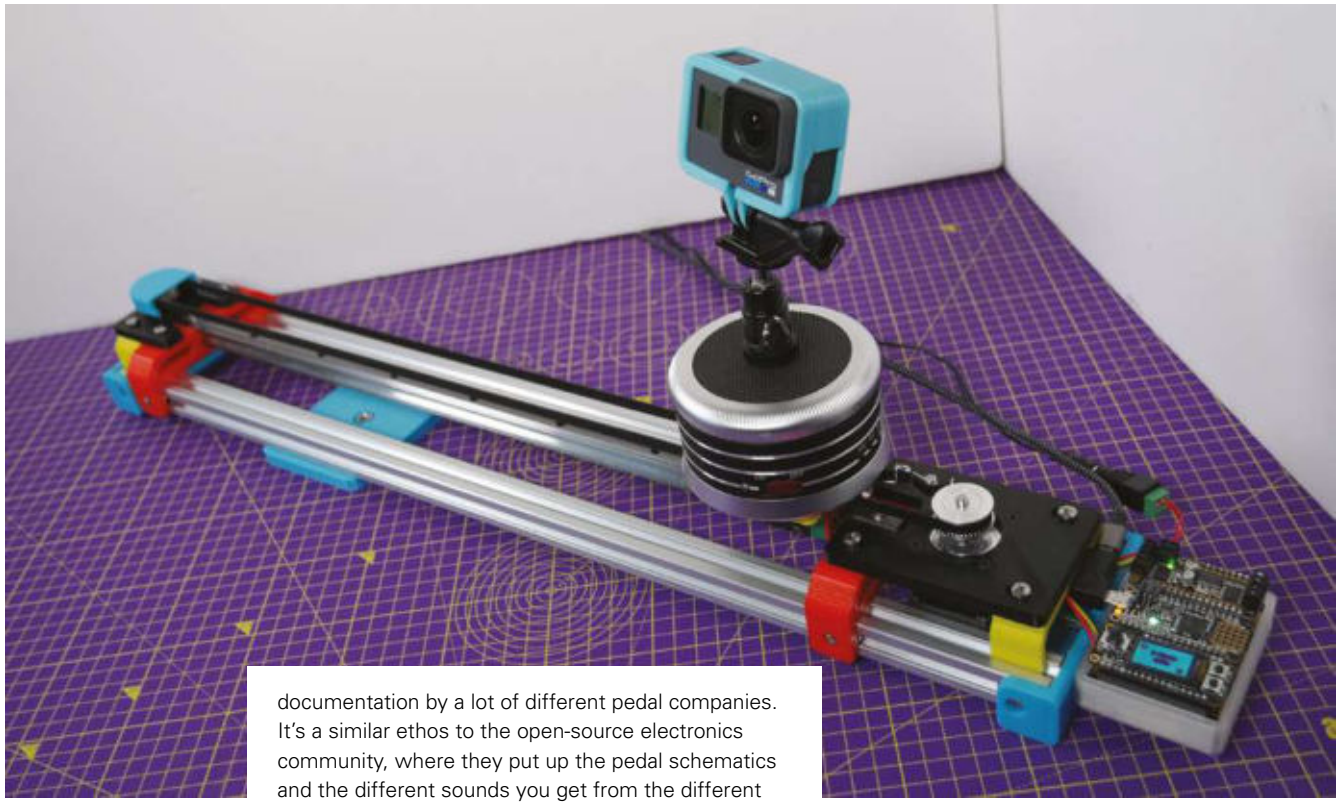
"I've done a few pedal teardowns, which I love because you get to see how stuff works. Once you learn a little about electronics, the experience of looking at a PCB moves from looking at all these bits on a circuit board that look artistic and interesting, to understanding how the circuit is working, because there's this chip and I know what this does, here's a component I recognise, and so on.

"I actually have some parts in right now to make a Fuzz Face, a classic 1960s guitar pedal. That's going to be a fun project. The Fuzz Face is also really excellent to modify; there are all these different things you can do with it. There's a lot of really good →



Left
The eyes of this 3D printed ghost are LEDs controlled by pulse width modulation

REGULAR





Top
Writing the official firmware for a product that's already on the market (such as this camera slider) is quite a challenge

documentation by a lot of different pedal companies. It's a similar ethos to the open-source electronics community, where they put up the pedal schematics and the different sounds you get from the different configuration, and they'll tell you how to breadboard it out.

"Open source is very important for me. I think it comes back to music; there's this kind of attitude and philosophy in the punk community called the DIY ethic, where you should be able to record your own band using whatever you can find, put your own shows together, and that's something I've always carried with me. When I found out there was a whole electronics and computer ethos with similar principles, I was really attracted to that.

"My first intro to coding was working with Arduino around 2011–2012 when I was in school, but I was totally on my own, and I was basically just copying and pasting code. I really didn't understand what I was doing – as I think with a lot of things when we start off, especially if you have no background in it growing up.

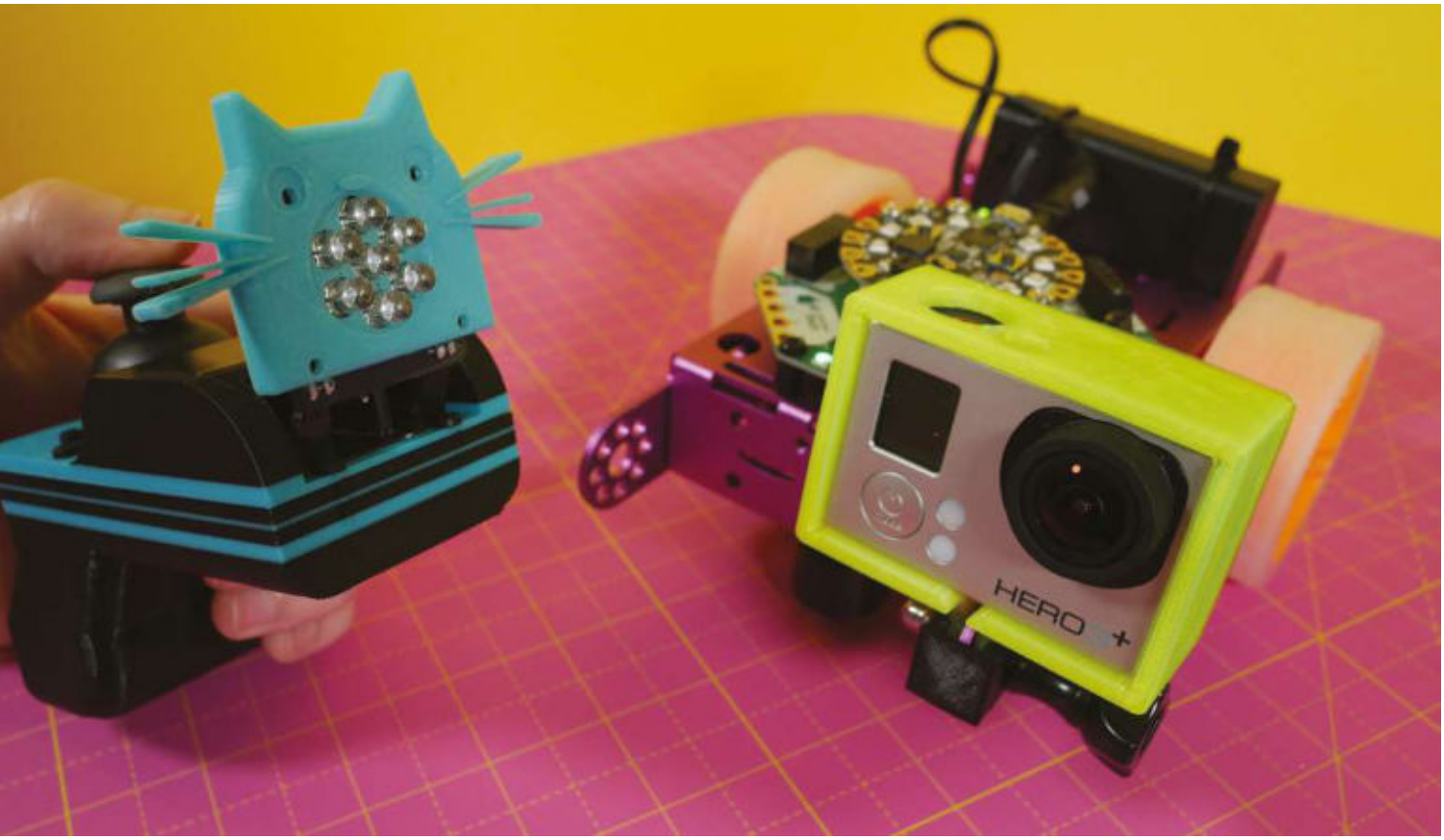
"Then, I started to get really serious about it, probably 2015–2016, and started to really understand what was going on with the code. The turning point was when I started working with CircuitPython and Raspberry Pi – that's when I started to get into it and to go deeper into projects, getting a better grasp of what coding is and how you do it.

> **The turning point was when I started working** with CircuitPython and Raspberry Pi

"Recently I've done a lot of learn guides for Adafruit; I really enjoy working with them, and I've done a lot of team-ups with Noé Ruiz, who is their 3D printing guru. He can just do things in Fusion 360 like it's nothing. We've teamed up on a few projects recently: a camera slider, a weather display, a MIDI guitar modelled after a Guitar Hero controller. Now he's helping me out with the Xylo Pi, which is a robotic xylophone. I think what they've done with CircuitPython is make coding really accessible to people like me who maybe don't have an educational background in it. The way that the libraries are set up is really easy to dive into and also to do more complicated things with the code.

CONNECTING BRAINS

"With Noé, when we work together we'll do a brain dump: take everything that we want it to do and put it in a document. For example, with the camera slider, we wanted it to have different sliding modes and have a menu that you could choose from: I worked on the menu bit first. And then I wanted to make sure that the timings were good, so I worked



on that, then put the two together and shared the code with Noé so that he can make sure it's working on his hardware. We'd build from there. The slider was a pre-existing project that we were bringing into CircuitPython. That was a unique experience because it already existed; I was putting my own spin on it a little, but also making sure that it fits the spec of what already existed so that it wasn't losing functionality.

"With the MIDI controller, we kind of thought of it in these two stages. We approached it from a hardware standpoint first. Firstly, I wanted to make sure that we could get notes out with the buttons on the neck, and then we also wanted a strum bar… you hold down the note, and the note wouldn't sound until you hit the strum bar, like hitting a string on a guitar. The first thing is figuring out if that's even possible. The next thing is making it work, and adding different modes using toggle switches. You grow it from there. We thought there might not be enough notes on the neck, so we added a rotary encoder. When you turn it, it will change what notes are on the neck.

"For this project in particular, we went for the hardware first, and also the aesthetic – how would it look, what would it be like to play, and then the code came after to make that work.

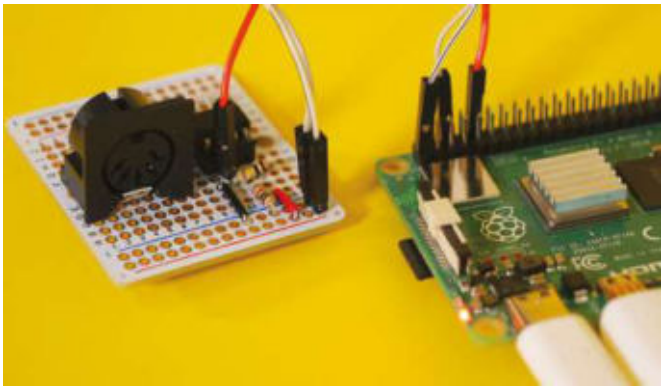
"We needed a controller with MIDI functionality, but also something that could emulate a guitar. It's set up by default to be chromatic, but the code is written so that you can have any note pattern. If you were using it to control a drum sequencer, for example, you could assign the buttons to do other things so it would be easier to play.

"When it came to learning how to design a PCB, I watched a lot of videos. Everyone has their own →

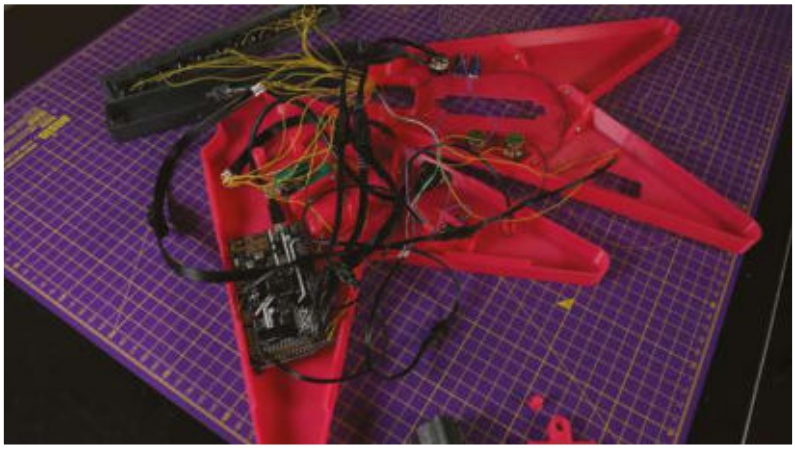


Below
A lot of Liz's builds combine a mix of technologies, such as this 3D printed enclosure for a micro:bit step counter

approach, and there are a lot of different programs you can use too. I use EAGLE because I found that the tutorials were more focused at beginners. Adafruit has a lot of great info on that; Ladyada does these great live streams going through the process of how she makes a PCB. I also learned a lot from Unexpected Maker, Seon. He does a lot of live streams using EAGLE to design PCBs. I think it's a great next step for people who are getting further into electronics and who want to get to more final-looking projects. Where something that maybe on a breadboard or perfboard would take up a lot of room, you can maybe shrink it down to the PCB because you can make the traces really short and everything. It feels like a proper next step to go to. Of course, there are mistakes that come with that, but that's half the fun.



Below
Liz collaborates on builds for Adafruit with 3D designer extraordinaire Noé Ruiz



"During the making of the Xylo Pi, I had a lot of problems with PCBs. With PCBs, a lot of stuff comes out that maybe you don't notice on the breadboard, because all of a sudden you're combining things together on to one board. I learned a lot about logic levels, and how you really need to pay attention to the difference between controlling a circuit with a Raspberry Pi and an analogue circuit.

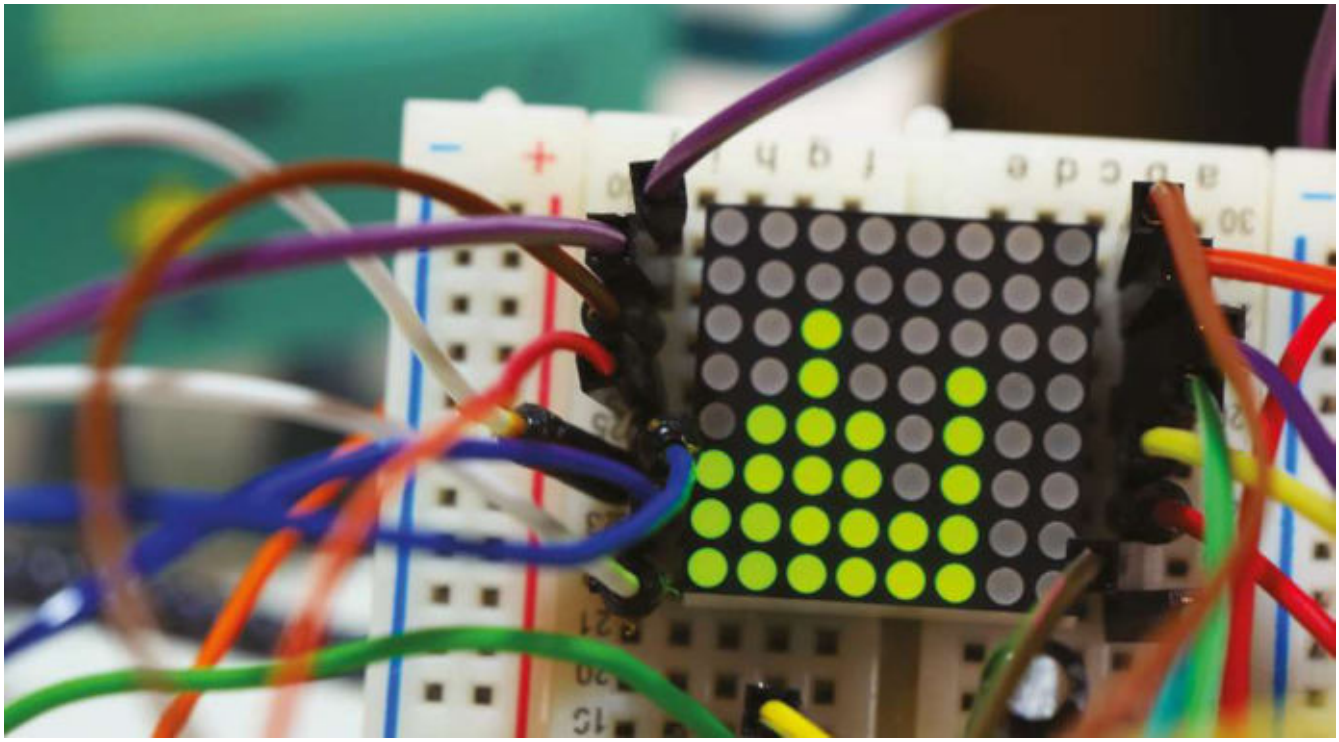
HATS OFF

"I've learned so much from PCBs when you're designing them. Trace width, how the signals are going to flow. It's a great learning process. The first one I made was a HAT for a Raspberry Pi. And it was just to take this little thermal camera module and



Left
Les Paul would have loved the innovation in this Flying V MIDI controller





have it so that it would just sit on the PCB, and it would be routed to the proper GPIO pins on the Raspberry Pi, so you could plug and play. I'd recommend doing something like that: have a little simple circuit that you want to connect to a dev board. That's an easy way to get used to the idea of routing traces.

"The book I wrote, *Practical Tinker Board*, came from the fact that I have a YouTube channel and had done a bunch of videos on Tinker Board. I was approached by Apress; they saw my videos, and asked if I wanted to write a book for it. I'd always wanted to write a book. Ever since I was a little kid of four or five years old, if you asked me what I wanted to be, I'd have said an author.

"And so I kind of approached it as what I wish I'd known when I started working with single-board computers and Linux. It is focused on the Tinker Board, but it also talks a lot about Linux, programming with Python, and all the operating system distributions that are available for single-board computers.



> **"I'm able to pick and choose things** that are going to be fun"

"I'm able to pick and choose things that are going to be fun. That's why, right now, I'm doing so much music stuff, because that's what's speaking to me. What's next? Whatever I feel like!" □

Left
Build instructions for this retro-themed IoT weather display are here: hsmag.cc/efSKgP. Why not make one for your DIY smart home?

Making with nature

Finding new interests while locked down



Lucy Rogers

@DrLucyRogers

Lucy is a maker, an engineer, and a problem-solver. She is adept at bringing ideas to life, and is one of the cheerleaders for the maker industry.

I enjoy working outside – one of my best holidays was making a Windsor rocking chair from green (fresh) wood in the woods. I prefer to take my chop saw out on to the patio when I need to cut stuff, rather than in the workshop, and if I can, my tea-breaks are taken outside.

I've always had a garden, but my gardening skills have mainly consisted of lawn mowing and 'taming', with a bit of intermittent weeding, rather than growing or planting things. The definition of a weed is something growing in the wrong place – I've weeded oak saplings, grass, and mint – all of which I'd have loved to have had in places where they are not! I did once try to create a living willow structure, where I planted willow whips into the ground. It became known as 'The Dog Basket' and was not very successful.

However, having recently been mainly restricted to my garden, and as the weather has been wonderful, I have started to take more of an interest in what is growing out there.

As garden centres have been closed, many seeds and plants online were sold out, so I wondered what I could hack. I vaguely remembered growing carrot tops when I was a child, so I placed my carrot trimmings in a saucer and was very satisfied to see them sprout. I have now planted them out. I have been most disappointed to learn that I can eat the carrot greens (leaves), but they won't grow another orange root.

I have also planted some seeds I found in the bird-seed mix I had. I didn't have plant pots, so I used old plastic milk bottles, with some drainage holes poked in the bottom. I have some 5 cm high seedlings now – which I am hoping will grow into sunflowers. Or maybe hemp.

> **I can eat the carrot greens (leaves), but they won't grow another orange root**

I've seen some great experiments online of people growing avocado stones, dried beans, and apple pips. I think when we are restricted in some ways, we crave being able to nurture something we do have some control over.

I saw recently that Sally Le Page did a YouTube video (**hsmag.cc/FHUnwe**) on making a wildlife pond – to encourage amphibians and insects and other mini beasts into the garden. I was impressed at the simplicity of her make – a pot, (could be an old washing-up bowl or a large flower-pot with no holes in it) some Hessian cloth and rocks – to help the beasties crawl in and out, some coir (coconut hair) and some pond soil and pond plants. Nice for a bird bath too. It's now on my list of things to make.

THE OFFICIAL **RASPBERRY PI**

CAMERA GUIDE

- Take pictures and shoot video with your Raspberry Pi and official camera
- Set up your HQ Camera or Camera Module and control it with precision
- Fun projects include a wildlife camera trap, security camera, and video doorbell
- 17 chapters packed with essential info for Raspberry Pi photography





Buy online: magpi.cc/cameraguide

Letters

ATTENTION ALL MAKERS!

If you have something you'd like to get off your chest (or even throw a word of praise in our direction) let us know at **hsmag.cc/hello**

COSPLAY

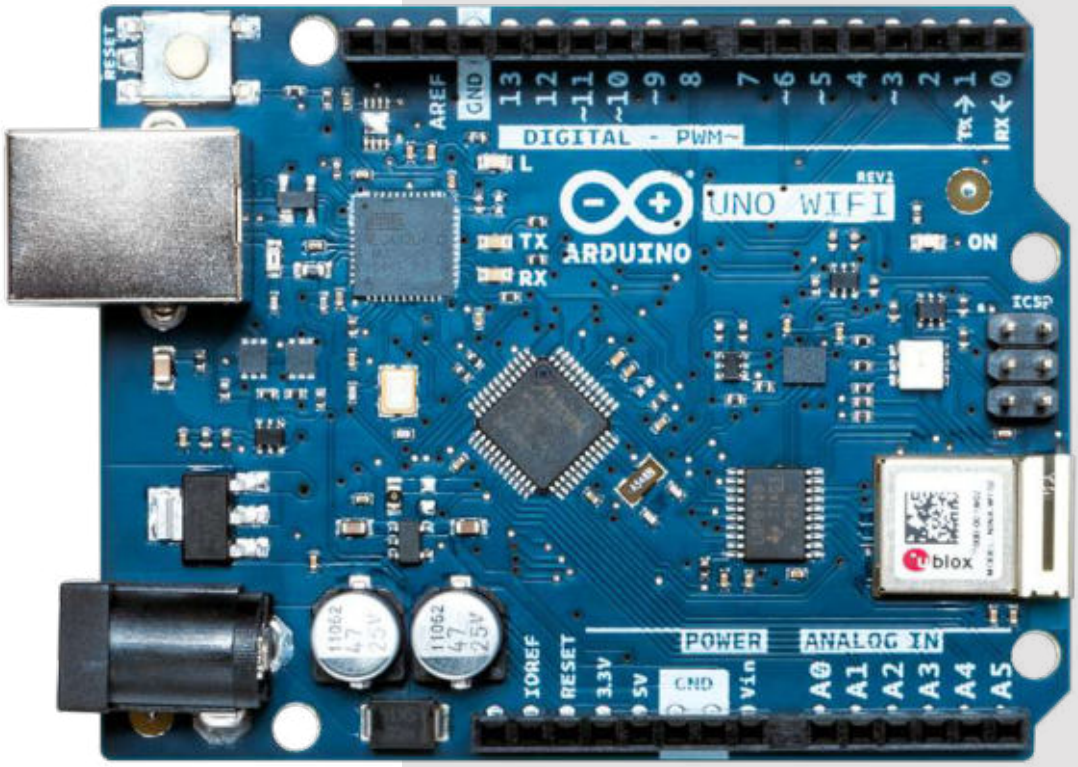
I really enjoyed your interview with the folks behind Kamui Cosplay in issue 30. It's great to see people do awesome things while still showing the challenges behind these creations. It's OK to not be an expert in everything, but in this social media-led world where we're bombarded with images of great creations in their final state, it's easy to forget this.

Trevor
Swansea

Ben says: We're indebted to Svetlana for her honesty in that interview. Trying, failing, and trying again is a huge part of being a maker, and it's a huge part of being a human. It's a natural thing to want to show off your successes, but if that's all we ever do, then we turn social media into a 'highlight reel' which can leave us feeling bad about ourselves when we inevitably can't live up to it. If you feel in a position to share your failures as well, then it's a great way to help others feel better about the challenges they face.







ARDUINO

I'm a programmer by day, and sort of fell into Arduino when I wanted to make a light that I could turn on and off. From there, I picked up a few more bits and pieces as I needed them, but I've never really taken a systematic approach to learning how to use microcontrollers. As such, I don't know what I don't know! Thanks for thinking of people like me, and helping us fill in some of the gaps in our knowledge.

Paul
Glasgow

Ben says: Glad you enjoyed it. The Arduino ecosystem's size is both its greatest strength and its greatest weakness. There's so much there to make great projects, but you have to be able to find it.

ULAB

I am the author of ulab, about which you wrote a very nice article in the latest issue of HackSpace magazine. First, thanks for bringing the library to a bigger audience, and giving it greater exposure, I really appreciate it.

Second, a couple of comments on speeding up the code:

1. Arrays support the buffer protocol, so you don't need to initialise a new array every time you have a full measurement: you can skip samples = ulab.array(samples_bit[3:])
2. ulab implements the argmax/argmin function, so you don't actually have to do the expensive linear search in python: index = arg_max(spectrogram) is also possible, and much faster. (By the way, you should break out of the loop, as soon as you have found the maximum.)
3. You don't have to calculate the log of anything, since your log is monotonic, and you just want to know the position of the maximum.
4. I am working on the implementation of interp, so the problem with the frequency resolution is going to go away.

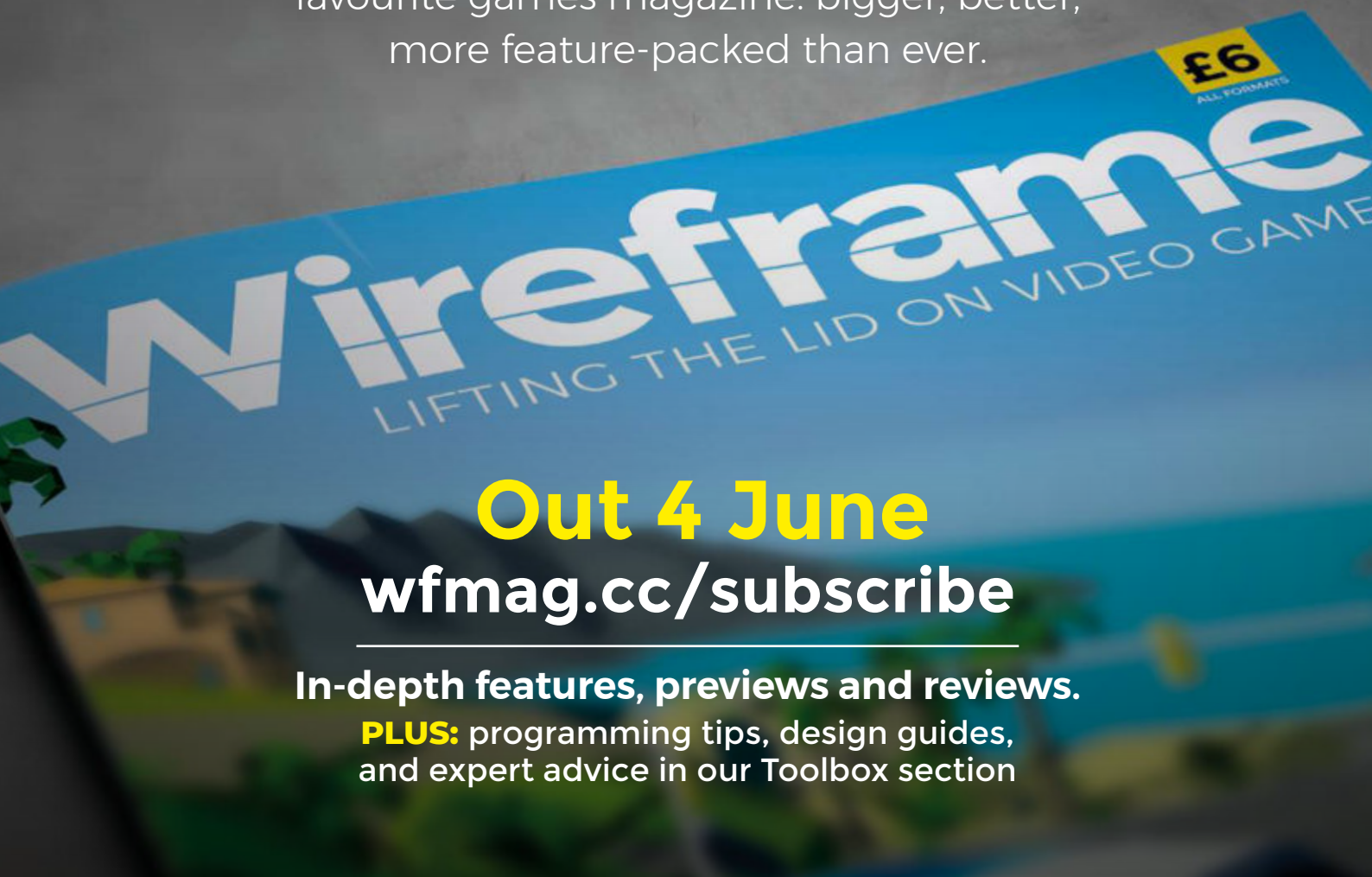
I believe, with these modifications you could gain a factor of two, perhaps even more in speed.

Zoltán

Ben says: Thanks for getting in touch. High-performance code has never been my speciality, so it's really useful to get feedback like this. Sometimes it's tempting to get a bit lazy and expect a library to do all the work on making your code fast, but it's important to remember that a library can only do so much. It's still up to you to make sure you use it in the best possible way.

BIGGER. BOLDER. MONTHLIER.

The new **116-page, monthly edition** of your favourite games magazine: bigger, better, more feature-packed than ever.



Out 4 June

wfmag.cc/subscribe

In-depth features, previews and reviews.

PLUS: programming tips, design guides, and expert advice in our Toolbox section

HackSpace
TECHNOLOGY IN YOUR HANDS

LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG 50

HOW I MADE: SEVEN-SEGMENT DISPLAY CLOCK

Salvaged parts, coupled with luxurious mahogany, to make a unique artefact

PG 54



IN THE WORKSHOP

Taking apart a car stereo to reuse an old DAB radio module – plus added plywood!

PG 56



INTERVIEW: KYLE WIENS

We talk to the founder of the world's largest repository of repair knowledge

PG 64

IMPROVISER'S TOOLBOX: PVC PIPES

Flexible, plentiful, and waterproof – these are a making goldmine

PG 36

DIY SMART HOME

Build your own Internet of Things

FEATURE

Build your own Internet of Things

By PJ Evans

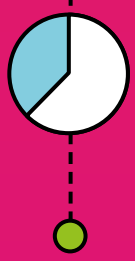


In the last decade, various companies have come up with 'smart' versions of almost everything. Microcontrollers have been unceremoniously crowbarred into devices that had absolutely no need for microcontrollers, and often tied to phone apps or web services that are hard to use and don't work well with other products.

Put bluntly, the commercial world has struggled to deliver an ecosystem of useful smart products. However, the basic principle behind the connected world is good – by connecting together sensors, we can understand our local environment and control it to make our lives better. That could be as simple as making sure the plants are correctly watered, or something far more complex. The simple fact is that we each lead different lives, and we each want different things out of our smart homes. This is why companies have struggled to create a useful smart home system, but it's also why we, as makers, are perfectly placed to build our own. Let's dive in and take a look at one way of doing this – using the TICK Stack – but there are many more, and we'll explore a few alternatives on page 46. →

Introduction and installation

Use your project's data to create alerts and beautiful infographics



Many of our projects create data, sometimes a lot of it. This could be temperature, humidity, light, position, speed, or anything else that we can measure electronically. To be useful, that data needs to be turned into information. A list of numbers doesn't tell you a lot without careful study, but a line graph based on those numbers can show important information in an instant. Often makers will happily write scripts to produce charts and other types of infographics, but now open-source software allows anyone to log data to a database, generate dashboards of graphs, and even trigger alerts and scripts based on the incoming data. There are several solutions out there, so we're going to focus on just one: a suite of products from InfluxData collectively known as the TICK Stack.

1. InfluxDB

The 'I' in TICK is the database that stores your precious data. InfluxDB is a time series database. It differs from regular SQL databases as it always indexes based on the time stamp of the incoming data. You can use a regular SQL database if you wish (and we'll show you how later), but what makes InfluxDB compelling for logging data is not only its simplicity, but also its data-management features and built-in web-based API interface. Getting data into InfluxDB can be as easy as a web post, which places it within the reach of most internet-capable microcontrollers.

2. Kapacitor

Next up is our 'K'. Kapacitor is a complex data processing engine that acts on data coming into your InfluxDB. It has several purposes, but the common use is to generate alerts based on data readings. Kapacitor supports a wide range of alert 'endpoints', from sending a simple email to alerting notification services like Pushover, or posting a message to the ubiquitous Slack. Multiple alerts to multiple destinations can be configured, and what constitutes an alert status is up to you. More advanced uses of Kapacitor include machine learning and anomaly detection.

3. Chronograf

The problem with Kapacitor is the configuration. It's a lot of work with config files and the command line. Thoughtfully, InfluxData has created Chronograf, a graphical user interface to both Kapacitor and InfluxDB. If you prefer to keep away from the command line, you can query and manage your databases here as well as set up alerts, metrics that trigger an alert, and the configurations for the various handlers. This is all presented through a web app that you can access from anywhere on your network. You can also build 'Dashboards' – collections of charts displayed on a single page based on your InfluxDB data.

4. Telegraf

Finally, our 'T' in TICK. One of the most common uses for time series databases is measuring computer performance. Telegraf provides the link between the machine it is installed on and InfluxDB. After a simple install, Telegraf will start logging all kinds of data about its host machine to your InfluxDB installation. Memory usage, CPU temperatures and load, disk space, and network performance can all be logged to your database and charted using Chronograf. This is more due to the Stack's more common use for monitoring servers, but it's still useful for making sure the brains of our network-of-things is working properly. If you get a problem, Kapacitor can not only trigger alerts but also user-defined scripts that may be able to remedy the situation.

Getting started

STEP 1
Choose your platform

The good news is the entire Stack is available for a wide range of platforms, including Linux, macOS, and Windows. Installation varies from platform to platform, but there are many articles on InfluxData's site to help you get started (**influxdata.com**). We're going to concentrate on the little powerhouse that is Raspberry Pi 4. You can run the entire Stack on a Raspberry Pi Zero if you wish, but, for best performance, the latest and greatest is a sensible choice.

STEP 2
Operating system setup

Start by choosing the right variation of the Raspbian operating system for your project. If you don't plan on showing video directly from Raspberry Pi 4, you may want to go with Raspbian Lite and save some memory and storage space, as all graphical interfaces to the Stack are web applications. Go through the usual steps of installing the OS, getting on the network, then make sure you're up to date with `sudo apt update && sudo apt upgrade` before proceeding further.

STEP 3
Prepare the repository

We start by adding InfluxData's repository key to our system using the command line:

```
curl -sL https://repos.influxdata.com/influxdb.key | sudo apt-key add -
```

All being well, you'll see 'OK'. Now add the repository itself:

```
echo "deb https://repos.influxdata.com/debian buster stable" | sudo tee /etc/apt/sources.list.d/influxdb.list
```

NOTE: This command is for Raspbian Buster. If you're using an earlier version, e.g. Stretch, replace the word 'buster' in the command with 'stretch'.

To read in the contents of the repository into our local cache:

```
sudo apt update
```

STEP 4
Installation

Now install the TICK Stack:

```
sudo apt install telegraf influxdb chronograf kapacitor
```

After a bit of work (might be time for a cuppa), your TICK Stack will be installed. Now start the database server and data analysis engine:

```
sudo service influxdb start
sudo service kapacitor start
```

It will respond quickly, but in the background, it will be setting up for the first time. Give it a couple of minutes. Assuming you're on the machine itself, the easiest way to check everything is working is to open a web browser and go to **http://localhost:8888/**.

If you're accessing remotely, find out the IP address of the machine and replace 'localhost' in the URL or try **http://raspberrypi.local:8888/**.

STEP 5
Initial configuration

You will now see a welcome screen. From here we can confirm that InfluxDB and Kapacitor are working as expected. Click 'Get started' to go to the next page, where we set up a connection to the database. Leave everything as default and click 'Add connection'. You'll now be taken to the Dashboards page. Click 'System', then 'Create 1 Dashboard'. Now we'll check that Kapacitor is working. Leave everything as default and click 'Continue'. If you get a success message, everything is working how it should be. Click 'View All Connections' to go to the main interface.

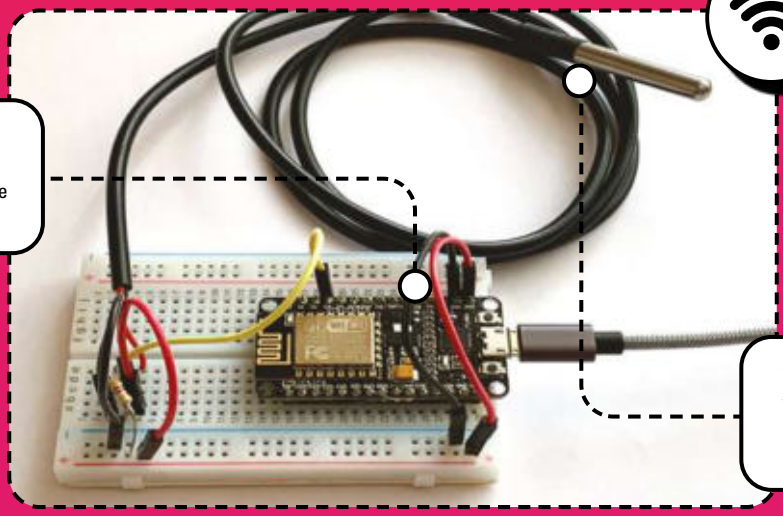
STEP 6
Your first dashboard

The previous step not only installed the entire Stack, but also set up Telegraf to start monitoring your local system, and created your first dashboard. To see it, click on 'Dashboards' on the left-hand menu bar, then 'System'.



Here you have a whole range of stats on your local computer's performance. We recommend taking some time to get used to the interface. Have a play with moving the panels around (you can add, remove, reorder, and resize to your heart's content) and adjusting the time ranges. Every panel can be edited to change graph type, adjust styles, and add thresholds. Just hover over the panel and click the pencil icon, then 'configure'.

You're all set to add your first data stream to InfluxDB. In the next tutorial, we'll build a simple logging device and use InfluxDB and Kapacitor to log data, display charts, and send alerts.



The ESP8266 is a popular WiFi-capable and Arduino-compatible microcontroller

The DS18B20 temperature sensor is cheap and accurate to 1/100th of a degree

Building our first sensor

Measure, record, and display the temperature

YOU'LL NEED

- TICK Stack running on a platform of your choice
- ESP8266 board (e.g. NodeMCU) or similar
- Breadboard
- 5 short jumper cables
- 4.7k resistor
- DS18B20 temperature sensor (we used the waterproof version)

STEP 1
What we'll be doing

Our fancy new TICK Stack isn't going to be very useful if we don't have any data to inject, so let's set up a simple project to generate something we can graph. The DS18B20 is a cheap and popular temperature sensor which comes in a standard TO-style package, or in a waterproof probe version, which we are using.

STEP 2
Set up the Arduino IDE

Firstly you need to set up your NodeMCU. We're using the Arduino IDE to program the microcontroller, so make sure it's set up and working for your choice of device (there are too many variants to cover here, but the internet is awash with tutorials). Once you're happy that your device can communicate with the IDE, you'll need to add in some libraries so your controller can support the 1-Wire protocol.

In the Arduino IDE, click on 'Sketch', then 'Include Library', and 'Manage Libraries'. In the resulting dialog box, search for 'onewire' and select 'OneWire library by Paul Stoffregen'. Next, in the same box, search for 'dallas' and install 'Dallas Temperature by Miles Burton'.

STEP 3
Get wiring

It's time to wire up your microcontroller. Make sure the USB connector is not connected as you do this. Study the photo here carefully and wire up the breadboard to match. As the 1-Wire name suggests, the DS18B20 has a single data wire and two further wires for power. The positive wire from the data sensor should be connected to the 3.3V pin on the NodeMCU, the ground should go to ground, and the data line should be connected to one end of the resistor and also to the data pin (D3). The other end of the resistor should be connected to 3.3V. Note that the data comes in as a digital signal, not analogue. When you've finished, check, double-check, and triple-check all your connections before reconnecting the controller back to your computer.

STEP 4
Check the sensor

Let's set up some code to test the temperature sensor. In the Arduino IDE, create a new project, and enter the



code – which you can download from **hsmag.cc/home1**. This sets everything up and reads the sensor once every ten seconds (you can change this by altering the value of 'interval' in seconds). Carefully enter the code (or just copy and paste it). If there are any issues, the compiler will let you know. When finished, click the arrow icon to compile and transfer the code to the microcontroller. Once complete, click the Serial Monitor (magnifying glass icon) to watch the output. You should see a list of readings from the sensor.

STEP 5
Create the database

It's time to shift our focus to the TICK Stack. We need to create a database for your temperature information. The easiest way to do this is from the command line. Open up a terminal on your computer (or use SSH) and access InfluxDB:

```
influx
```

We'll create a database for our sensor:

```
CREATE DATABASE room_monitor
```

Don't worry if it doesn't return any results – that means it worked. One of the great things about InfluxDB is that you don't need to create tables or define schemas; you can just start adding data. Try this:

```
USE room_monitor
INSERT room_temp value=20
SELECT * FROM room_temp
```

Here you've switched to our room monitor database, adding a temperature reading of 20º and then queried the data. See how the time stamp has been automatically added for you? You can have as many tables as you like, so you could add light readings, humidity, or pressure. Just do a first `INSERT` command to create them automatically. To leave the shell, type 'exit' followed by return.

STEP 6
Start sending data

Let's bring the two parts of our project together. First, we need to get our microcontroller on the local network, then start sending data to the InfluxDB installation. You'll need the name (SSID) and password of your WiFi network and the IP address of your InfluxDB installation. Download the code from **hsmag.cc/home2**, replacing the `<values>` in the variables to match your local settings. Now compile and replace the code on your microcontroller. Open up the Serial Monitor to make sure everything is working as expected. The controller should connect to WiFi and send a message to InfluxDB every ten seconds.

All being well, you'll see output on the console similar to this:

```
Waiting 10 seconds
room_temp value=19.62
204
```

The '204' means 'successful, but I have no content to give you', i.e. 'it works'!

STEP 7
Create a graph

Let's check on the data coming into InfluxDB. Open Chronograf in a browser and click on the 'Explore' icon. To the left, you'll see a list of databases. Click on 'room_monitor.autogen' to show the metrics belonging to that database. Click 'room_temp', then 'value'. You'll now see the beginnings of a chart monitoring your temperature data. The Explore section is a real playground for the nascent data scientist. You can add multiple queries or click on the 'Visualization' tab to change or style your graph.

STEP 8
Create your dashboard

Have a play with your graph until you've got something with which you're happy. Now click 'Send to Dashboard' in the top-right corner. Select 'Send to a New Dashboard' and give it a name, such as 'Temperature', then name the Dashboard 'My Room'. Click on 'Dashboards' on the left, then select 'My Room'. You have the beginnings of a dashboard.

STEP 9
Be alert!

Finally, let's add an alert. For this to make any sense, you need something to receive the alert. We used Pushover, a great API for generating smartphone notifications. Go to **pushover.net** and sign up for a week's free trial, and install the Pushover app on your phone. When you log into the Pushover website, you'll see your user key. Make a note of this. Now scroll down and 'Create an Application/API Token'. Fill out the short form, and you'll be rewarded with an API token. Make a note of this also.

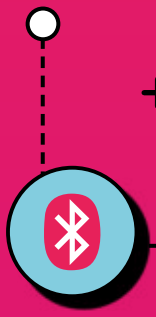
Back in Chronograf, click on 'Alerting' then 'Build Alert Rule'. Give your rule a name, select 'Threshold' for the type, then select 'room_monitor', 'room_temp', 'value'. For conditions, set a temperature for the alert threshold. We went with 25 as we can simulate that by holding the sensor. Now, under 'Alert Handlers' click 'Add a Handler' and select 'pushover'. You'll be directed to the handler configuration page. Select 'Pushover', then enter the two keys and make sure 'Configuration Enabled' is checked. Save the changes, then go back to the alert rule and reselect Pushover as the handler. You'll now be able to set a message, and you can use variables too. We entered:

```
It's  {{ index .Fields "value" }}ºC!
```

Click 'Save Rule'. If you've got everything up and running, grab hold of the sensor so that the temperature goes over the threshold. Your phone should spring to life!

Advanced TICK Stack and showcase

Advanced TICKing



We've shown how you can use the TICK Stack to monitor a remote device that is generating data, but we've barely scratched the surface of what's possible. Let's look at the next steps that you can take in your data visualisation journey.

You can have as many databases as you like

InfluxDB, like any other database server, allows you to have any number of databases. We've created two in this feature: the Telegraf database that monitors the machine, and 'room_monitor' which houses our temperature sensor. A single server can be used for multiple applications, and InfluxDB runs well, even on smaller hardware like Raspberry Pi computers.

Each database can have multiple measurements

In a time series database, what would normally be called 'tables' are known as 'measurements'. Each database can have multiple measurements acting as a logical group. So, our 'room_monitor' could have any number of relevant (or not, we won't judge) measurements added and you can change them at any time.

Each measurement can have multiple values

If your sensor or device produces multiple measurements per reading, it's easy to add those readings in a single entry. For instance, you may choose to have multiple temperature readings from different sensors on the same microcontroller added at the same time. This is ideal when you want to easily compare different readings. You can also add 'tags' to each reading, which are like labels that are not part of the reading but allow you to record additional metadata.

You can manage data retention automatically

Every 'measurement' in InfluxDB has a retention policy attached. This is to prevent your disk filling up with data you may not need anymore. If you want to know what your CPU temperature was this time last year, that's great, but chances are you don't, so you can have InfluxDB automatically remove data that's older than a given value, say one week or a month.

You have a built-in API server

As you've seen from the tutorial, InfluxDB exposes a very simple API that allows you to write and query the database by making HTTP POST requests. Here's an example:

```
URL:
http://192.168.0.20:8086/write?db=room_
monitor

POST Data:
room_temp value=20
```

Calling 'write', you specify the database on the query string, then the measurement followed by the reading in the POST body. Full information here: **hsmag.cc/vSUrPx**.

Alerts can get very clever

If a simple alert rule isn't enough for you, there is an entire JavaScript-style scripting language, TICKscript, that allows you to create complex rules. So if you want an alert when the temperature goes above a certain value, and it's the first Thursday after a full moon, that's entirely possible. You can also create custom alert plug-ins and trigger multiple alert services per alert event.

Introducing Grafana

There's a related component we haven't introduced yet. Grafana is made by a different organisation, but is certainly a close relation of the TICK Stack. Grafana is an open-source project that concentrates on making dashboards and graphs similar in functionality and appearance to Chronograf. The big difference is its ability to use many different data sources, not just InfluxDB.

To get the latest version of Grafana on Raspberry Pi (and other Debian platforms), run the following commands:

```
echo "deb https://packages.grafana.com/oss/deb
beta main" | sudo tee -a /etc/apt/sources.
list.d/grafana.list
wget -q -O - https://packages.grafana.com/gpg.
key | sudo apt-key add -
sudo apt update
sudo apt install grafana
sudo systemctl enable grafana-server
sudo systemctl start grafana-server
```

Just like the other applications, it's entirely web-based and runs on port 3000 by default. So, to access, make sure you know the IP address of your server and go to:

http://<IP Address>:3000

(Replacing <IP Address> as needed.) You'll be asked to log in (use admin / admin) and then to change your password. After that, you can add your local InfluxDB data source. Click on 'Add Data Source', then enter the following values:

```
URL: http://localhost:8086
Database: room_monitor
```

Leave everything else as default and click 'Save & Test'. Now click on 'Dashboards', 'New Dashboard', and 'Add Query'. Change the 'select measurement' box to read 'temp', and a graph should appear. You can now experiment with styling and type just as with Chronograf. Grafana can analyse data from many different sources, including SQL databases, logging systems, and cloud services. If you can't see the one you're looking for, don't worry: there's a huge community offering free plug-ins to add extra data sources, and if you're handy with a bit of code, you can build your own as well.

The TICK Stack is designed to be as accessible and usable as possible, so its default is no security at all, although there's plenty to be added if you want to lock down your services. This is not the case with Grafana, which features security on by default. It's a great choice for business applications with full-screen kiosk modes to create attractive dashboards for monitoring essential services. That doesn't mean you can't have fun with it at home, of course.

If you're producing graphs that may be of interest to others, many services publish their Grafana dashboards publicly, and you may want to join in the fun (see the 'Remote Access' box above). So, whether you're a data geek who likes nothing better than looking at the latest stats from CERN, or want to tell the world about the current light levels in Basingstoke, you can share as much (or as little) as you like. →

Remote Access

If you want to be able to see your dashboards remotely, you can consider opening a port of your local router to forward to your Grafana installation. There are security concerns with doing this, so do your research first. Inbound VPN solutions for Pi-hole and WireGuard are more solid choices, or use Grafana's cloud service to publish your chart remotely (chargeable).

Beautiful dashboards

It may come as little surprise that both Grafana and the TICK Stack are used all over the world for critical monitoring. The result is often some beautiful and inspiring dashboards. Here, we'll have a look at some great examples from the showcase at **hsmag.cc/eM3V8l**.

GitLabs › dashboards.gitlab.com



Popular source code hosting service GitLab attests to its 'openness' policy by showing all its service data publicly

CERN › monit-grafana-open.cern.ch



CERN's Kubernetes cluster is a beast. Yes, that's nearly 30TB of memory alone

Wikimedia › grafana.wikimedia.org



One of the busiest sites on the planet – you can watch the Wikipedia Foundation's network services in real time

Security Warning

Grafana comes with a solid security model that is enabled by default. The TICK Stack has similar capabilities, but without anything enabled. That means from the point of installation, anyone can write to your InfluxDB just by knowing the IP address. If you're doing anything on a public network, be sure to investigate the user and permissions capabilities and lock everything down.

GridKa › grafana-sdm.scc.kit.edu



This data centre in Karlsruhe, Germany, deals with output from a little machine called the Large Hadron Collider. You may have heard of it...

Other options

If the TICK Stack doesn't fit your needs, here are some other options

We've looked at the TICK Stack here for building a home monitoring system with Raspberry Pi and Arduino-compatible boards, but there are many other options for linking together sensors and actuators to make your home system.

Let's take a look at some other popular options. These are each based around the idea of having a central server and connecting devices to it. You can use almost any computer for the central server, but Raspberry Pi boards work well in all cases as they're small, low-power, and don't have any moving parts (that can become clogged with dust if constantly running).

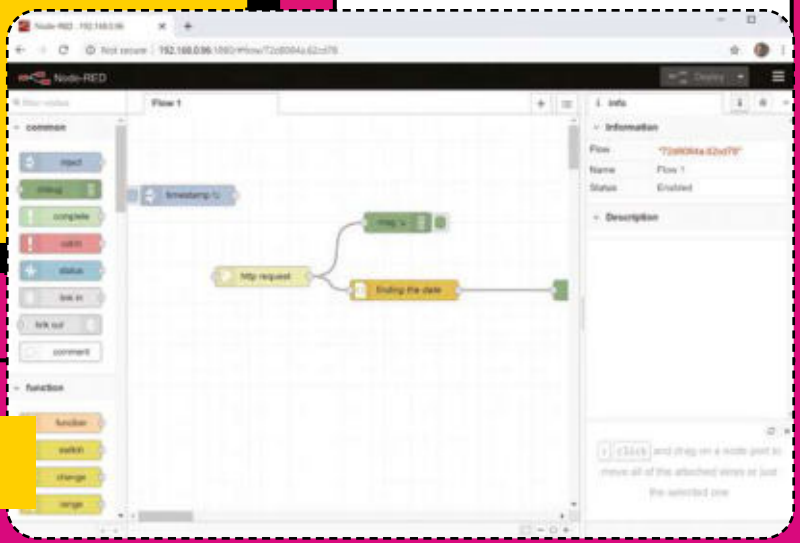
With each of these, you need a way of communicating with your input and output devices. They all provided interfaces for makers to use with their own code, but if you're planning on using commercial smart home hardware, it's worth checking that there's a way of linking the software to the hardware before committing to a plan.

Node-RED

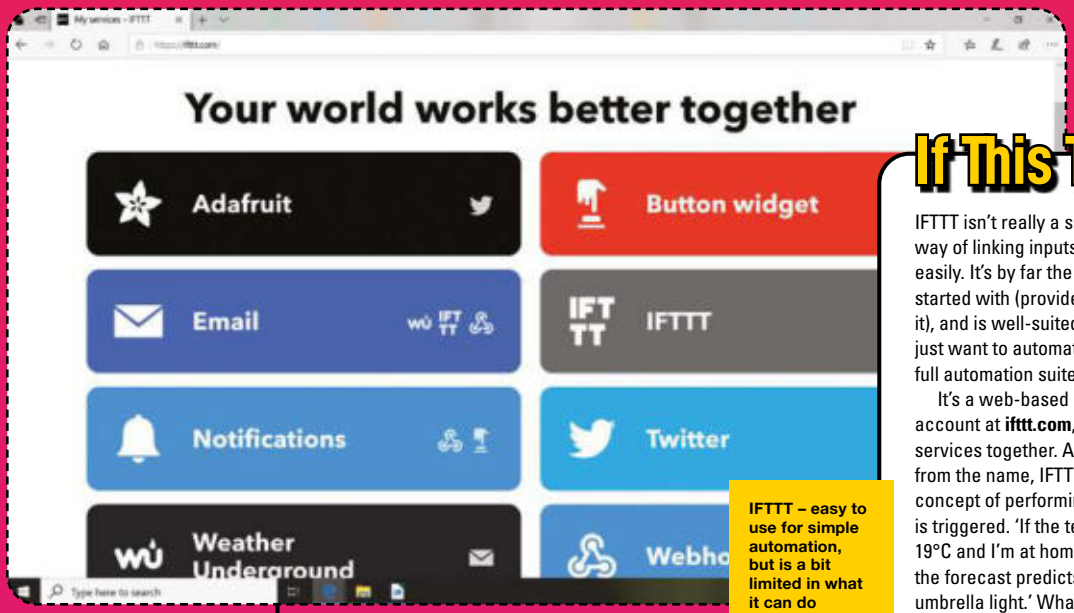
Unlike the other options, Node-RED isn't really a home hub, but a programming environment that's well-suited to applications like smart homes. Once installed, it hosts a web-based development environment that you can access from any computer. In this, you can drag and drop 'nodes' into your workspace, and link these nodes together to create flows. These flows are paths that data can move through your program, and the nodes define how it's processed as it goes. For example, you could link a motion sensor and the time of day to a node that decides whether or not to turn a light on. This node could output to a controller for a smart light.

Don't be too put off by the fact that it's a programming environment – you can create your entire program by dragging and dropping elements, so it's suitable for people with no programming experience.

Because it's not designed for smart homes, Node-RED is incredibly flexible, so you can link it to far more services and get it to perform more tasks than the other options.



Flow-based programming in Node-RED



IFTTT – easy to use for simple automation, but is a bit limited in what it can do

If This Then That

IFTTT isn't really a smart home setup, but it's a way of linking inputs and outputs together very easily. It's by far the easiest option to get started with (provided your hardware supports it), and is well-suited to occasions where you just want to automate one thing, so setting up a full automation suite is overkill.

It's a web-based service, so you need an account at **ifttt.com**, and you can link different services together. As you may have guessed from the name, IFTTT is built around the concept of performing an action if a condition is triggered. 'If the temperature goes below 19°C and I'm at home, put the heating on.' 'If the forecast predicts rain today, light up my umbrella light.' What it does do, you can set up quickly and easily, but it's quite limited, so you might find that you just can't do some of the things you want to.

OpenHAB and Home Assistant

These two bits of open-source software have a lot in common. They're both designed to link together a wide range of smart home devices, and provide the user with a simple interface for managing their home. They both have an automation framework for applying a set of rules to automatically control how your home works, and you can create broadly similar home automation setups using either piece of software. Our advice, if picking between them, is to first check the hardware you want to use. If only one supports it, then use that. If they both do, have a look at the rules engine, as they're a little different and one will probably seem a bit more comfortable to you than the other.

HackSpace

When not to use TICK

We're showing a lot of love for the TICK Stack here, but it doesn't mean it's a hammer to crack all nuts. The Kapacitor data analysis engine is at the heart of things and it is primarily mathematical in nature. If the data you produce is inherently non-numerical in nature, it may struggle to understand. For example, a data stream of colour names isn't easily usable, but the RGB values of each one could be processed.

Voice Assistant

Alexa, Google Assistant, Mycroft, and others are voice-controlled interfaces that let you interact with the digital world with speech. You can add your own 'skills' to them that tell them how to interact with your hardware. If you build a device that's controlled over HTTP, you should be able to link it to your digital assistant so you can manipulate it with speech. Results can vary, depending on how complicated it is to interact with your smart thing, but for some uses, these can make digital devices really easy to use.

SUBSCRIBE TODAY FROM ONLY £5



SAVE UP TO 35%

Subscribe today and get:

- **FREE delivery**
 Get it fast and for FREE
- **Exclusive offers**
 Great gifts, offers, and discounts
- **Great savings**
 Save up to 35% compared to stores

Subscribe online: hsmag.cc/subscribe

SUBSCRIBE TODAY

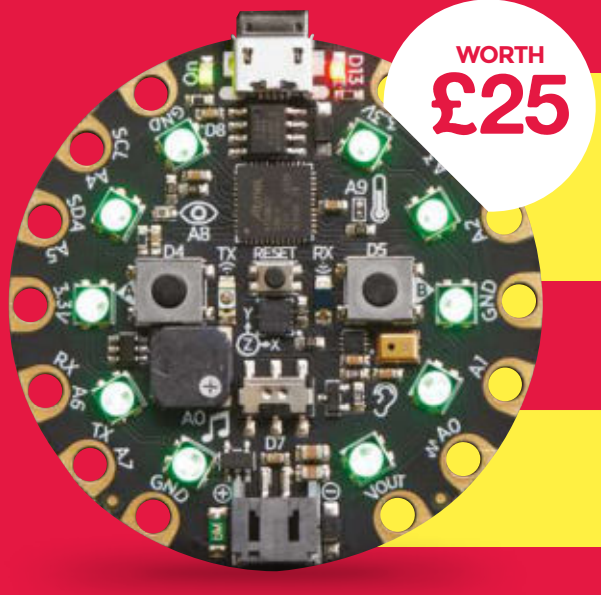
Subscribe for 12 months | **Rolling monthly subscription**

£55 (UK) **£90** (USA)

£80 (EU) **£90** (Rest of World)

Free Circuit Playground Express with 12-month upfront subscription only (no Circuit Playground Express with rolling monthly subscription)

- **Low monthly cost** (from £5)
- **Cancel at any time**
- **Free delivery to your door**
- **Available worldwide**



FREE!

Adafruit Circuit Playground Express

With your 12-month print subscription

This is a limited offer. Offer subject to change or withdrawal at any time.

SUBSCRIBE on app stores

From £2.29

Available on the App Store

GET IT ON Google Play

Buy now: hsmag.cc/subscribe



FEATURE

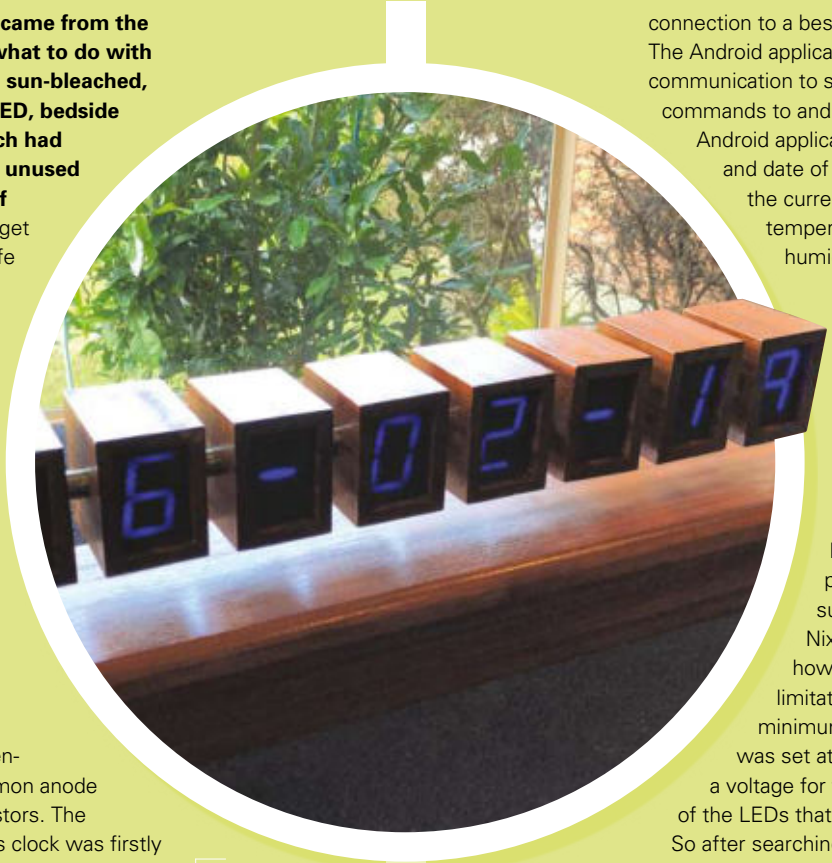
How I Made
WIFI SEVEN-SEGMENT CLOCK

Build your own time teller from old equipment

By Christine Thompson

This project came from the question: what to do with two cheap, sun-bleached, four-digit LED, bedside clocks which had languished unused in a drawer for a couple of years? The answer was to get creative and breathe new life and style into these two clocks, give them a fresh appearance, improved functionality, and lastly, have fun with the challenge of making it. My projects have always been a combination of electronics, (although I am the first to say my electronics knowledge is hovering just above zero), woodworking (cutting a straight line is a challenge), and finally software (bugs, bugs, and just more bugs).

Where to start? The seven-segment clock uses a common anode 5V supply via 220ohm resistors. The main reason for building this clock was firstly the reuse of two bedside clocks each with 4×7 segment displays, but also the inclusion of a WeMos D1 R2 board which allowed the connection to a bespoke Android application. The Android application uses WiFi communication to send and receive commands to and from the clock. The Android application can 'SET' the time and date of the clock and can 'GET' the current time, date, temperature, pressure, and humidity. It was at this point that I ran into a problem which required a complete rethink of the project, namely the intention was to use the very capable IC Chip MAX6921 to power all eight LEDs. This chip had been used on previous projects to successfully power both Nixie tubes and other LEDs, however, I found that its one limitation was that its minimum output supply voltage was set at 8V, which was too high a voltage for the 5V maximum voltage of the LEDs that were going to be used. So after searching for solutions with Mr Google, and after asking a question on a very good Google Group called [Neonixie-l], I had some great help from David at the Nixie



Above
The finished time piece in action

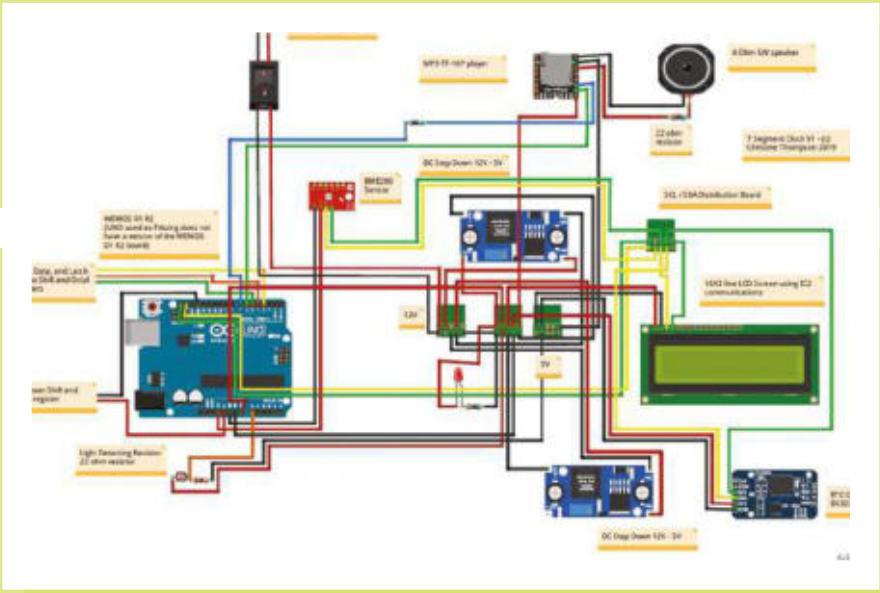
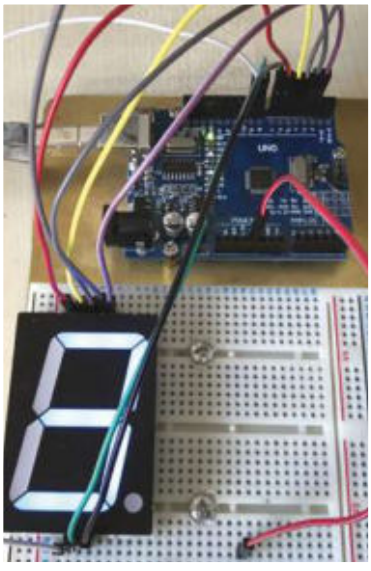


Figure 1
The wiring setup for the clock

Google Group who kindly provided me with a schematic (the hand-drawn image is his great work) of a suitable 74HC595 SPI 16 shift register and a 74HC245 octal tri-state transceiver register-based circuit to support the 8×7 segment LEDs using the multiplex method of display. A simple PCB board was constructed using two 74HC595 20-pin IC chips located on 20-pin carriers, and two 74HC595 16-pin IC chips located on 16-pin carriers. The output of one side of the circuit was used to support the anodes of each of the 8×7 segment LEDs, and the other side of the circuit was used to support the seven segments plus the decimal point, via 22 ohm resistors in series.

My shopping list included:

1. WeMos D1 R2 Arduino card with on-board ESP8266 WiFi module
2. Light-detecting resistor plus 22 ohm resistor
3. Two-pole switch, coloured wires, PCB female plugs, heat shrink, PCB board, 3 mm plastic supports
4. LED plus 330 ohm resistor
5. BME280 temperature sensor
6. MP3-TF-16P player plus 22 ohm resistor
7. 4 ohm 5 W speaker
8. 16 × two-line LCD screen using I²C communications (optional, used mainly for testing)
9. RTC DS3231
10. 2 × DC step-down 12 V – 5 V
11. 2 × 74HC245 IC chip plus 20-chip carrier
12. 2 × 74HC595 IC chip plus 16-chip carrier
13. 8 × 22 ohm resistor



Left
Testing out one of the seven segment displays

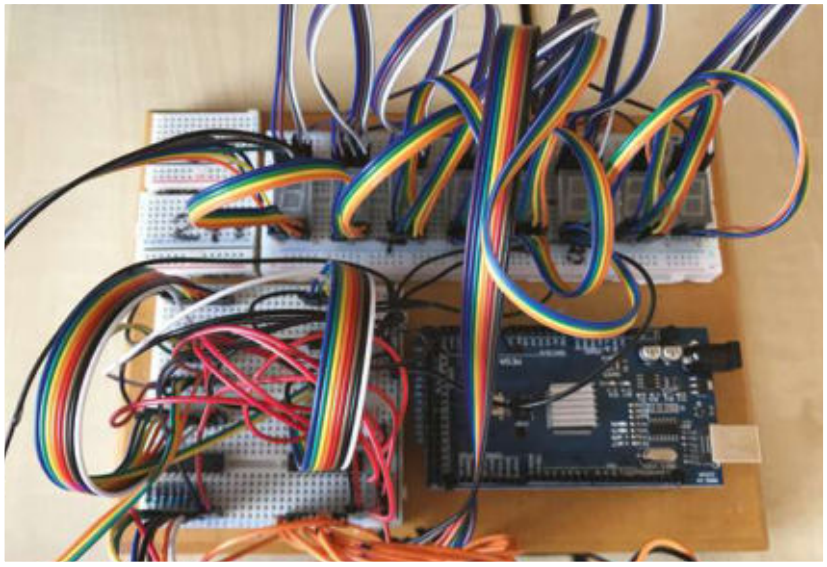
CONSTRUCTION

The clock case was constructed from mahogany with eight simple boxes constructed to surround each of the seven-segment LEDs. Each box is connected to the next using a 15 mm steel tube which passes through each box, and via a hollowed-out mahogany cube which turns the steel tube through 90 degrees. The steel tube is fixed to the hollow box below which contains the clock support equipment. The wires connecting each LED are fed through each box and via the steel tube down to the clock system below.

The various photos show the layout of the basic components onto the clock's baseboard. The use of a distribution board for both the I²C communications and 5 V power has the advantage of only requiring two pins on the WeMos board, and also allows two DC-DC step-down 12 V to 5 V supplies to be used: the first to supply power to the board, LCD, RTC, MP3 player, etc.; the second dedicated to powering the clock display and display driver circuit. The photos attached to this article show some of the steps taken to build the clock housing, its enclosed LEDs and wiring, and the circuit as laid out on the 5 mm baseboard.

Two of the photos show the initial breadboard-based design of the circuit where a single LED, (seven segments, each with two LEDs within each segment) is powered via an Arduino Uno and the second shows the completed system, powered by an Arduino Mega, using the shift registers to power eight simple seven-segment common anode LEDs.

The baseboard circuit shows a 16×2 LED display which actually is not necessary; however, it was included as it shows independently that the system is working



Left
It's not as complicated as this wiring looks

> This area of the project can be improved greatly by removing the reliance on wiring, and replacing it with a suitable PCB board

One product that helps me with this and other projects is Fritzing. While simple, it allows for the layout and design of simple circuits. The first schematic (**Figure 1** overleaf) shows the design of the clock in general. The second schematic (**Figure 2**) shows the design of the shift-based registers and how they were attached to the LEDs.

It's at this point that I feel that I should own up to my lack of abilities. The images show the construction and wiring of the circuit – this area of the project can be improved greatly by removing the reliance on wiring, and replacing it with a suitable PCB board. Currently, this is way beyond my abilities, although I am currently learning how to design and create a simple PCB board using KiCad.

without the need to resort to the WiFi option. As with all of my projects, I use off-the-shelf modules and pre-built 20 cm DuPont wiring. A 4 ohm 5W speaker was attached to the MP3 module, which allows the clock to chime on each quarter of the hour. The MP3 player uses a single 1GB microSD card which contains a number of MP3 files – four containing the quarter-hour chimes, and one containing a single-hour chime sound recording. One interesting point that I found was that the Arduino code must have an in-built delay after each play chime command is made to the MP3 player, otherwise, no sound is heard. The RTC DS3231 provides time and date and is supported, while power is off, by a single CR2032 button battery. The BME280 temperature sensor is a great compact sensor which provides temperature, pressure, and humidity readings. The circuit includes a light-detecting resistor, plus a 22 ohm resistor circuit which provides a night-time, display-saving feature which is especially useful with clocks that are Nixie tube-based. Finally, a single-pole switch, power adapter connector, and LED were attached to the circuit so that they could be mounted to the outside of the clock case. The entire clock is powered by an off-the-shelf 12V 1A power adapter.

SOFTWARE

Two programs were developed: one based on the Arduino IDE platform, and the second using Android Studio to develop the Android app. The ICO file contains code which allows the WeMos to control the BME280, RTC, and LCD screen. The WeMos D1 R2 Arduino software was based on a previous WiFi-based robot that I had created, where a WiFi communications

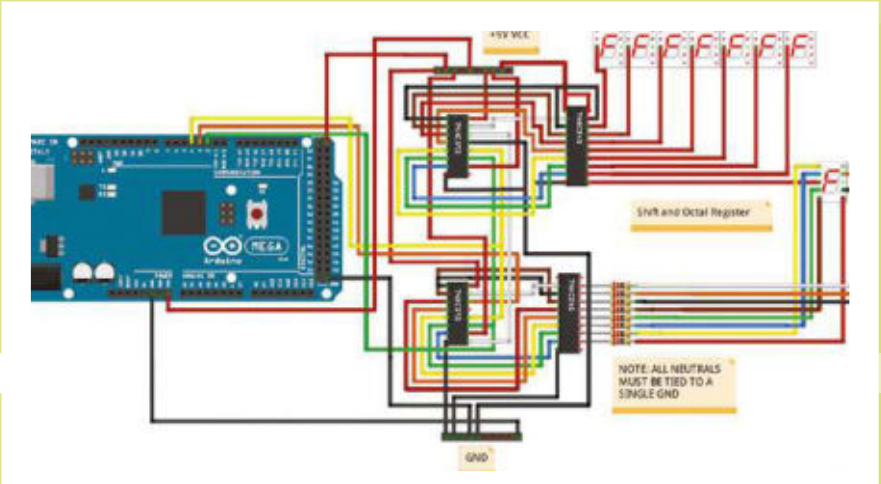


Figure 2
Using Fritzing for design



Right
I love the look of the wooden boxes

package was added using simple 'GET' and 'SET' host commands to 'GET' the robot's on-board video, and 'SET' the direction of movement the robot would travel in. The Arduino software was developed using version 1.8.10 Arduino IDE, and the board selected was the 'LOLIN (WeMos) D1 R2 and Mini'. It was necessary to download and install the following special libraries: Wire.h, LiquidCrystal_I2C.h, SoftwareSerial.h, DFRobotDFPlayerMini.h, SparkFunBME280.h, RTClib.h, ESP8266WiFi.h, WiFiClient.h, and ESP8266WebServer.h. The WiFi Access Point created by the WeMos ESP8266 chip is called 'WifiClock' and has a password of 'password'. It's possible to update the clock not using the bespoke Android app, but rather simply using a standard web page viewer, with the 'WifiClock' access point selected, and entering the following HTTPS command:

For the SET command:

```
"http://192.168.4.1/SET?PARA1=HH-MM-
SS&PARA2=DD-MM-
YY&PARA3=VV&PARA4=Y&PARA5=Y"
```

Where time and date are entered using the standard format and 'VV' is the 0–30 chime volume; the first 'Y' next to PARA4 is 'Y' or 'N' to select the chimes to be played option, and the second 'Y' next to PARA5 is 'Y' or 'N' to select the night-time display-saving option which closes down the display during the hours of darkness. For the GET command:

```
"http://192.168.4.1/GET"
```

This returns a string of data from the clock in the following format:

```
HH,MM,SS,DD,MM,20,YY,HHH,HH,
PPP,PP,CC,CC,FF,FF,VV,Y,Y
```

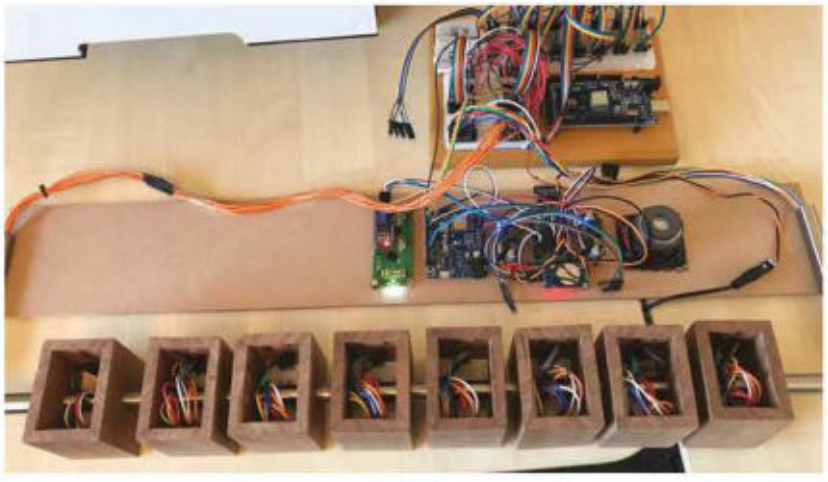
Where 'HHH,HH' is the humidity reading; 'PPP,PP' is the pressure reading; 'CC,CC' is the temperature in centigrade; 'FF,FF' is the temperature in Fahrenheit; 'VV' is chime volume; 'Y' is chimes required, and the second 'Y' is night display-saving required.

The development of the Android Studio-based application went hand in hand with the clock's development, and was largely based on the previous work I'd done where the application could send commands and receive video from a remote WiFi-based robot. The application consists of two screens. The first screen has buttons to 'GET' and 'SET' clock values, and a further set of two buttons to 'GET' the tablet's current time and date. The second screen – accessed via a WiFi icon button from the first screen – provides access to the WiFi listing, its associated password, and various other tablet settings.

It is important to ensure that the tablet's location services are enabled; otherwise, the WiFi scan button will not return any available networks, including, of course, the WiFi clock network.

This project has brought together two new elements, namely the use of WiFi as a method for updating the clock, rather than the use of an in-built keyboard. Secondly, the use of a shift register and octal register-based control circuit for the seven-segment displays. I find great satisfaction in reusing old redundant equipment and bringing it back to life. The development of an Android-based application allows the clock to be viewed remotely, although a 20-metre range limit is all that can be expected from the WeMos ESP8266 chip and its limited power. An alternative to the shift-based display driver that I have used is one using the MAX7219 IC display driver chip which is designed to provide the 5 V supply to seven-segment-based displays.

The author freely accepts that their work may be considered amateurish; however, this and other projects of mine satisfy a need – that being to create something that is aesthetically pleasing to the eye, functional in its nature, challenging to design and construct, and most of all, great fun to build. If you are interested in seeing other work by the author, it can be viewed here: **hsmag.cc/dQOMn0**.



Left
Final step: hide the wires!

IN THE WORKSHOP: A DAB radio and Bluetooth speaker

By **Ben Everard**

Taking apart a car stereo



Above
I'll be honest, I didn't expect a lot from these speakers (especially given how they're mounted), but the sound quality's not bad. Not great, but perfectly acceptable for a prototype

Last month I converted an old radio into a WiFi speaker, and this month I experimented with another technique for creating a wireless speaker.

A little while ago, I bought a cigarette lighter-powered DAB radio for my car, but due to the position of the handbrake, couldn't actually fit it, so that got me wondering. I've been investigating building a DAB radio into a pair of vintage speakers for a while, but it's surprisingly hard to find a hackable DAB module at a reasonable price. Rather than seeking out a module, why not pull this radio to pieces and see what I can make out of it? I'd already attempted to fit it, so I couldn't return it anyway.

Unlike a WiFi speaker, this one will need a user interface, so it's going to require cutting holes in the speakers. Before I took the saw to the vintage speakers, I decided to make a prototype using a few bits I had sitting around for another project (a pair of speakers and an amp that will hopefully one day become part of a synth).

The unit has a leg that should slot into a car cigarette lighter port, then a body with an LCD screen, a scroll-wheel, and four buttons. Surely it's possible to take the leg off and mount the screen on another panel? With my screwdriver, I set to work finding out.



Above
Don't rush with your jigsaw or you'll end up with horrible cut-outs like these

With the back of the case removed, I noticed that the power connection was soldered onto the footprint of a USB connector. Car cigarette lighters give out 12V, so this seemed a bit odd. On closer inspection, there was a 12V to 5V conversion circuit in the leg. This was obviously designed to be something else before it became a car radio. Fortunately, I noticed this before pushing 12V into the unit.

I wanted to create a prototype to make sure everything worked, but I also wanted it to be usable because an important part of the prototype process is making sure it fits in with your usage. There's no point hacking apart the vintage speakers if it turns out I don't really listen to DAB and just use the Bluetooth.

I had a square of spare 9mm plywood, so I used this to make a panel. Designing an acoustically efficient speaker enclosure is a complex process, but I decided to sidestep it entirely and simply not have a back to the panel. This compromises the audio quality, but not enough for me to worry about for a prototype.

The task of building the prototype was then just to cut out appropriately sized holes in the panel, mount everything, and wire it all up: nothing a drill, jigsaw, and Dremel particularly struggled with. I've always liked the idea of exposed electronics on things, so decided to mount the amp circuit board on the front and show it off. Again, this was another little experiment to see if I liked living with exposed electronics. As it was just a prototype, I didn't put in quite as much design work as I should have, and there are a few flaws – the aerial cable goes under the bottom of the panel, and this means it doesn't sit flat, and the audio input hole is too close to the amp. They're all things I can live with for a prototype, but ones I'd change if it were a permanent design. Note to self: think more about cable routing in the future.

Since the amp needed 12V and the radio needed 5V, I decided to power the system with 12V and use a step-down converter to get 5V for the radio (I would have mounted this like the amp, but I didn't realise that I needed 5V until after I'd created the panel, so this went on the back).



The major problem with using a repurposed car radio is that it wasn't designed to be panel mounted, so there are no mounting holes. Since this is a temporary build, I didn't want to do anything permanent to either the PCB or enclosure, so went straight for the hacker's best friend: hot glue. While the speakers and amp are held with screws, everything else is kept in place with hot glue. If I progress on this build, I'll do something a bit more secure – possibly 3D-print a bracket to hold it in place, or use a bit of epoxy to stick it on permanently.

With everything hooked up, I powered it up and tested it out. The DAB to amp to speakers all worked flawlessly (although the amp is a little overpowered for the speakers, so you have to be a bit careful with the volume knob). Press the scroll button and the DAB radio turns off, turning it into a Bluetooth speaker. My phone (a Moto G7) has a bug in the Bluetooth audio system, such that even the maximum audio over Bluetooth is really quiet, so having an overpowered amp makes this usable as a Bluetooth speaker for me.

Overall, I'm pleased with the repurposed car radio as an audio source. It has an intuitive interface, works well, and the audio line-out jack is easy to use with other audio systems. I'll keep it running like this while I think of the best way to mount it on my vintage speakers. □

Above
Testing out the position of the parts

Below
One blob of glue had to be visible in order to hold the front cover of the radio unit in place



HackSpace magazine meets...

Kyle Wiens

Say hello to the MacGyver of the internet

One of the perils of making things is that eventually you will end up breaking things too. Sometimes it's on purpose, to salvage a part; sometimes because, well, things just break. If you've hit your broken thing with a hammer, checked YouTube, and asked the neighbours to take a look and it still doesn't work, you need something more: you need iFixit.

Since its humble beginning as a site hosting a single repair manual, iFixit has grown to be a huge repository of knowledge, a community, and a guide to fixing all manner of things. It also does advocacy work, fighting for the right to fix the things that we own. We spoke to Kyle Wiens, founder of iFixit, about why and how he started it, what ownership means, design, and a whole load of other things, including the second law of thermodynamics. →

Above
“The internet of things is the internet of outdated Linux distros”, says Kyle.

HackSpace Let's begin at the beginning. What made you start iFixit?

Kyle Wiens I was trying to fix my laptop, it was an iBook that I had dropped on to the power cord. I knew that it was a loose solder joint inside the laptop, so I started trying to take it apart. I realised very quickly that there were loads of tabs and latches, and it was very hard to understand how to do it. I googled "how do I open this thing?", but I couldn't find any information: there were no videos, no service manuals, no manufacturer information, nothing.

I went ahead and did the repair, got it working, but the computer was never quite the same – I broke some tabs and latches in the process. I know that Apple had a service manual. I knew what it looked like – I just couldn't find it online. I did a little sleuthing online and found that Apple had sent legal DMCA takedown notices to anyone who had posted their service manuals online.

I said: 'This is bull!!', wrote a new manual and put it online. And the rest is history. We've been doing it ever since. If we can't have the manufacturers' manuals, then we'll make our own.

HS Right, so that's how you get around the copyright restrictions that some manufacturers use to stop people sharing repair knowledge: it's your original work.

KW At first we were writing our own and publishing them, then we built a wiki platform so that anyone can come along and share information. That's really kicked off, and now we have over 60,000 how-to guides, for everything from skateboards to kitchen mixers to cars… you name it.

HS What sort of push-back have you had from manufacturers?

KW It's interesting because, on the one hand, we have sort of a symbiotic relationship. We're helping the customers make their products last longer, so we're outsourced customer support for them. On the other hand, we're definitely undermining any planned obsolescence strategy that the marketers might have. So it depends.

When it comes to a legislative perspective, where we're working on policies that will structurally change the economy, the manufacturers are very aggressive in opposing us. Apple is out in front at doing everything they possibly can to stop right to repair. But if you talk with individual Apple employees and ask them what they think of iFixit, they all use it. Jony Ive's design team uses our tools.

> We have over 60,000 how-to guides for everything from skateboards to kitchen mixers to cars

HS Legislation – as in, right to repair legislation?

KW There are right to repair laws on the books in the US in Massachusetts that are applied nationwide for automobiles. So we basically have the right to repair for cars, up to a certain point.

The EU also has some right to repair laws, and so does Canada, so probably you can get information to work on cars. When you're talking about tractors, you go to a farmer and say, 'I can replace the transmission on my Ford F150 pickup truck – I cannot do that same repair on my current John Deere tractor, because I don't have the software to do it.' Physically he can do it, but he doesn't have the software to enable it.

That's the same with the iPhone; physically we can do it, but we don't have the software to enable the new part.

And we know, as engineers, that you can use cryptography to limit what people can do. You can cryptographically tie parts to the phone, which is the Touch ID button, the sensor, that's one of the parts that we can't repair. It's cryptographically tied to the main board at the factory, and you need their key to sign a new part.

HS That kind of makes sense…

KW There's no reason why you can't, as the owner of the device, sign a new part, right? You should be able to authorise and precision your own parts. Cory Doctorow likes to say, if someone puts a lock on a device that you own and doesn't give you the key, they're not doing it for your benefit.

HS They would argue I guess that stopping people modifying the software on the operating system of a tractor or a car stops the user from driving too fast, hurting themselves, or violating emissions laws. Is there any validity to that?

KW What are the limits of agency? What do we take personal responsibility for? You can take the brakes off your car and then drive it down the road. Is that a good idea? If you do that, is it BMW's fault when you crash into a building? There are lots of things that we can do, and we have legal frameworks for managing that kind of situation. Just because I am using software to remove the brakes rather than physically taking them off, does that change fundamentally what I did? I don't think so.

It's the same thing with emissions: can you use software to bypass emissions controls? Yes. Can you go over and remove the catalytic converter and bypass emissions rules and increase your fuel efficiency? Yes, you can. Is it John Deere's fault if you remove the catalytic converter? No. At that point, it's between the farmer and the regulatory agency. →





Above
iFixit started because of a broken Apple laptop; it now hosts over 60,000 repair manuals

Right
It's not all bad news: repairable brands in home tech include Dell, Lenovo, HP and Motorola (this is a Motorola Razr)

Fundamentally this is all about control. Who has control? Is it the owner, or is it the manufacturer? And in all these conversations that the manufacturers have with legislators from the EU and the US, they say basically that the manufacturer is the one who has all the information; they're the one we should trust. They're the ones that should make the decision. And we're saying no, it's the owners who should make the decision.

I kid you not, the global automaker group, before the VW emissions scandal, they told us at a DMCA hearing in the US, they said that the systems in these vehicles are very complex, and the only people who know how to manage and moderate them safely are the manufacturers. No one should be allowed, not just to modify it, but to inspect or do security research, any kind of analysis on these systems. This was two months before Volkswagen got caught.

It's fundamentally a question of control, and the manufacturers are going to want control, and over time, if we allow them to have it, these devices will be tied to the cloud, and you'll be allowed to do precisely what the manufacturer wants and no more.

Imagine that we live in that world and fast forward 50 years, 100 years. Who are the next tinkerers? Everyone who's reading this grew up in a world where you could take hardware apart and put it back together again. But if your children cannot and their children cannot, and the world is locked down, over time you're going to see fewer engineers and tinkerers because there's less opportunity to take things apart and understand how they work. And so, you're going to see a pipeline of new engineers and tinkerers by virtue of the world being locked down.

We see this already simply with the industrial design of the iPhone. At science fairs and Maker Faires, we run a take-apart booth and we'll ask kids, 'hey, do you want to take apart this iPhone?', and they look at us like we're crazy, because every adult their whole life has always told them not to take things apart. These lockdowns for short-term profit maximisation are, in the long term, going to end up very costly for them because it's going to strangle the pipeline of engineering talent down the road.

HS I guess that won't hurt the bigger companies as much because they can train people in-house. It'll hurt competition coming up though.

KW We certainly see that in software. Amazon and Google have sucked up a lot of software talent in the world, and it makes it harder for software companies to access to talent.

> "'Hey, do you want to take apart this iPhone?', and they look at us like we're crazy"

HS What are you working on right now campaign-wise?

KW We're working on hospital equipment. We're building an open







database of service manuals for every piece of hospital equipment out there. We've got the NHS list of all the devices they use, and we're working on other lists, and we're trying to get to a comprehensive set of service documentation for all current hospital equipment.

We've been working with the Electronic Freedom Frontier on it, we have a team of about 200 librarians and archivists, who are helping us, and we're very close. We've already got the most comprehensive set of service manuals online for ventilators, anaesthesia machines, and dialysis machines, and we're working on expanding it to all hospital equipment.

HS Are you getting any resistance from manufacturers?



KW There's a petition to get the manufacturers to release the information themselves, and one manufacturer said that they didn't think that this information is something the world needed to have. In this case, we're not doing teardowns; we're publishing the manufacturer service manuals. The challenge really is organising and curating and making the information available in one place. We're not going to teardown a £1 million MRI machine. Though that would be fun!

> We're not going to teardown a £1 million MRI machine. Though that would be fun

HS If you were starting out now learning how to tear things down, where would you start? If iFixit weren't around?

KW Everybody starts by fixing their own stuff. Get as far as you can before you get stuck and you need help. Then the best way to learn is to find someone who's an expert at it do it with you – that's absolutely the best way to learn to do anything. Then, how do you learn on the internet? Forums are generally the best way to go.

HS So there's no one product out there that's the Rosetta Stone of teardown?

KW Start with something you have that's broken. If you fix it, great, if you don't, you're no worse off than you were when you started. This approach to life is pretty universal across products. Whether you're tinkering with your car, your laptop, or your bicycle, that approach to tinkering and learning is the same across all products.

HS We're running a feature on building your own smart home in this issue. Do you have any advice on how to do it right?

KW There was one home hub called Revolv, I think, that was a British company that got bought by Nest. After Nest bought them they shut down the servers. Well, that shut down everybody's houses. You had sprinkler systems tied to people's home hub, and when they shut off the servers everybody's sprinklers stopped working, their doors stopped locking, everything – it's just crazy.

There's so much that goes into building a resilient system. If you think about the electricity grid, it's incredibly robust and resilient. And a big part of that is that it's decentralised and it's decoupled, and if your start meter stops working, that doesn't matter, because there's a fallback. It doesn't need to talk to the network in order to keep supplying you with electricity. Consumer electronics designers are just missing the boat entirely when it comes to designing for resilience. If you design a product that requires internet access to function, you've failed. Everything should degrade to dumb mode.

You want to make a smart speaker, fine – put an aux-in jack on it. If the internet goes away or if you shut down your servers, it may not be a smart →

speaker anymore, but it will still be a speaker. That should be a fundamental design goal for anyone building anything 'smart'.

There's this car rental company in the Bay Area, with an app that you can use to unlock the car and rent on demand. So these reporters rent a car, and they go camping. As soon as they leave cell phone reception range, when they shut the car off, they couldn't get back. The car wouldn't start because they had no phone reception. In what design case does it make sense that a car requires constant cellular connection in order to function? This is crazy!

It's interesting, there's a fundamental set of tendencies that very large manufacturers fall into. It's the same pattern over and over again, whatever industry it is, whether you're making iPhones or tractors or medical equipment, we see manufacturers follow the same pattern: they start out friendly towards consumers, they have repair options, they start out supporting their products for a long time, and then the financial engineers take over and realise that there are other ways they can make money, by limiting how people use their products.

I think that may be part of why the home tinkerers, we're all still in the early stages of manufacturing and innovation. The dystopian vision is that if you get successful and your company is big and at the point that you've got 1,000 employees, can you preserve the ethos that you had in the beginning?

HS I guess, to use the John Deere example, tractors were never one source to begin with though.

KW With most products, it's not about that. Fundamentally, what do you need in order to fix something? You need information, you need tools, and you need parts. You are always going to have a manufacturer service option, and you are always going to have consumers – owners of the product – doing their own repair or hiring local professionals to do it.

When you have a manufacturer with a profit and loss statement attached to the service network that they're running, and they say, 'How can we grow that?', the tendency is to see all their servicing as a fixed pie, and the question becomes, 'How do we gain market share?' An obvious answer is that they can limit access to third-party repairers.

When I talk to technicians at hospitals, they'll say that they ask manufacturers for the repair manuals for equipment that they own, and the manufacturers refuse on the grounds that they think the hospital will share it with the local service guy, who will use it to compete with the manufacturer.

There's this zero-sum game mentality that leads manufacturers down the road of cutting off access to the information, parts, and tools that we need to repair things.

> They all make parts available. Power tools are an interesting shining ray of hope

Think about how many people buy new phones because their battery isn't happy. We saw this when Apple got caught two years ago slowing down phones with older batteries; everyone realised, 'Oh gee! I can just put a new battery in my phone, and it'll be good as new!' It turned out that it did, and it had a material impact on Apple's earnings. Tim Cook said that the dip in Apple's earnings that year was because people realised that they could put a new battery in their old phone.

HS That's crazy! That's like a Streisand effect, but for batteries.

KW You talk with their executives, and they would view that as a mistake, but it was like, all of a sudden, the wool had been removed from people's eyes and they understood.

Batteries are really insidious because they fail slowly, and we as humans aren't good at visualising that. We rationalise failing performance away as being down to a new software update, or the phone being old, or a lot of things. The phone's not wearing out; it's just a consumable. Throw a £20 battery in, and you're good to go.

The main thing for us now is that it's all about the community. iFixit is not me; it's not my knowledge. There are pockets of repair knowledge around the world; they're just not evenly distributed. It's our job to bring people together and distribute that knowledge. And do it in a curated, systematic fashion. There's a lot of amazing repair information knowledge on YouTube, but it's lost in among all the clutter.

At the moment, we're adding around 10,000 guides a year. We just added a couple of thousand power tools. That's another big project right now: we want to be really systematically comprehensive on power tools. They fail – but they're highly repairable. With a power drill most people, if the chuck goes out, they would throw it away, but they're super-repairable. And the manufacturers generally design them to be repaired. They all make parts available. Power tools are an interesting shining ray of hope, that the industry, in general, supports its products very well, but consumers don't necessarily know about it.

HS Finally, you've said that you're trying to fight the second law of thermodynamics. What does that mean?

KW We're fighting a war against the universe! We make things, then the moment you make it, it starts breaking. The only way that society stays functioning is because we are fixing things faster than they break down. □

Above
“Power tools are a shining example of a type of hardware that’s easy to fix. Give it a go – if your drill is broken, the worst that will happen is that you’ll still have a broken drill”

FEATURE

IMPROVISER'S TOOLBOX

PVC PIPES

A piping hot tool



Mayank Sharma

@geekybodhi

Mayank is a Padawan maker with an irrational fear of drills. He likes to replicate electronic builds, and gets a kick out of hacking everyday objects creatively.

Polyvinyl chloride, more popularly known as PVC, provides the plumbing that keeps our world connected. As one of the most used synthetic plastic polymers, PVC is used extensively in the fields of construction, engineering, healthcare, and technology to provide the ideal medium for everything including water, cement, intravenous medications, and bits and bytes as they to flow from one place to another.

Considering our dependence on the thing, it's surprising to know that it was discovered accidentally – not once but twice, first by French chemist Henri Victor Regnault in 1835, and then decades later by German chemist Eugen Baumann in 1872. Both of them identified PVC as a polymer, but failed to patent their accidental finds.

> "Although it is a plastic polymer, since PVC can be reshaped at very high temperatures, it is possible to reuse and recycle it"

PVC spent a considerable amount of time inside labs, primarily because, in its rigid form, it was impractical for any real-world application. It wasn't until the 1920s, when other types of plastics were added, that PVC was pliable enough to be moulded and put to use. Still, it wasn't until 1932 when the first tubes made from a PVC copolymer were produced.

Almost half of the world's annual PVC production is used to create pipes, mostly for industrial and municipal uses. As with many products, the evolution of the manufacturing process of the PVC pipe has had a great impact on its evolving usage in the last century. The first mass production method involved melting a powdered form of PVC before rolling it out. Although viable, this method was cumbersome and expensive. It gave way to the ram extruder method, in which the melted PVC is pushed through a mould to give it a uniform shape. In the 1960s, chemists found new ways of making PVC more flexible, which led to a tremendous rise in its adoption across different fields.

Although it is a plastic polymer, since PVC can be reshaped at very high temperatures, it is possible to reuse and recycle it. To recycle PVC, it is usually broken down into small chips and then refined to its original pure white form after removing the impurities.

Just like polystyrene foam, PVC is quite cheap worldwide, and widely available in all sorts of configurations. This is good because, unless you are a plumber, replicating the builds in the following pages will require a visit to your local hardware store. PVC is rigid, but can be cut/shaped by hand tools. Remember to always wear a respirator and eye protection when working with PVC pipes; sawing or cutting them produces minute particles that can do real harm if you breathe them in.



DREMEL DRILL-PRESS

Above
The build is quite taxing, but Johnny has done a nice job of explaining the process and taking extensive photographs of each and every step

So, you have a Dremel and now you need a drill-press to mount it on. You can either get one from the hardware store, or follow Johnny's lead and build one yourself using PVC pipes at a fraction of the cost. He begins by fixing his Dremel 4000 on one side of a 5"× 6" board with two 1.5" conduit clamps, before gluing a PVC bush and a small pipe on the other. He then uses several tees and pipes to assemble the base frame. The process isn't cumbersome thanks to the self-explanatory pictures.

After preparing the base, he attaches the 20" and 11" PVC pipes for the Dremel to slide on. Before putting in the springs that he's sourced from a 4" sprinkler body, Johnny suggests you sand them a bit for a fluid movement. Now come the real cumbersome bits: creating parts for the slide section, the handle support, and the handle slides requires a lot of cutting, sanding, and gluing. As a bonus, he's also created a rubber stopper and a plug that can be used to fasten the Dremel at one point. This enables him to use the Dremel to do some routing and sanding tasks. →

Project Maker
JOHNNY BAI

Project Link
hsmag.cc/ISHxum

FEATURE

HYDROPONIC UNIT

Project Maker
NATHAN WILLIAMS

Project Link
hsmag.cc/3mBiLA

Nathan has been interested in hydroponics since even before he was old enough to understand what was really involved in that kind of farming. A software engineer by profession, he first dabbled with hydroponics farming by growing a chilli plant. Buoyed by its success, Nathan decided to build himself a proper hydroponic garden with PVC pipes and some other bits. He uses recycled materials, including old PVC pipes, an old wooden baby's crib, and other miscellaneous junk, but you might have to source them from the hardware store. You'll also need a small submersible pump (Nathan uses an 18W one), and some plastic cups for your plants. The construction is fairly easy to replicate, thanks again to Nathan's explanations and detailed photographs of the process. He begins by making holes for the cups in the PVC pipes using a drill and a Dremel. Nathan suggests you space the cups depending on what you plan to grow in them. You'll have to follow his instructions to then prepare the cups, and how to run a hose through the PVC caps before fastening them at the end of the pipes.



Below
Nathan's build is based on the nutrient film technique in which nutrients flow over the roots continuously, bringing it lots of oxygen and food

> "THE CONSTRUCTION IS FAIRLY EASY TO REPLICATE, THANKS AGAIN TO NATHAN'S EXPLANATIONS"

If you don't mind putting in the effort, take a look at Paul Langdon's IoT-enabled hydroponic farm (**hsmag.cc/MfFIUC**).

BOOK/TABLET HOLDER

Project Maker
SPECIFIC LOVE CREATIONS

Project Link
hsmag.cc/jUdi5b

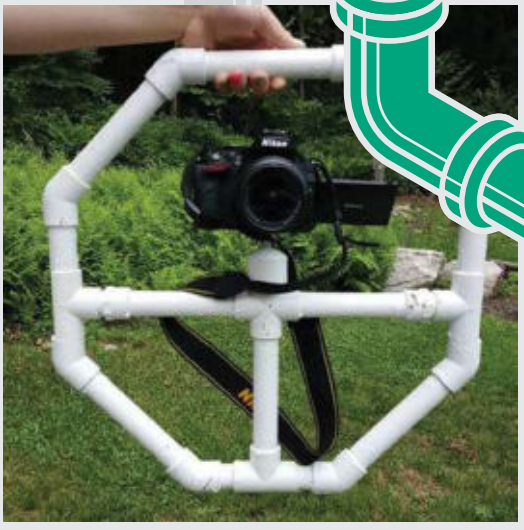


Right
In addition to holding a book or a tablet, it makes for a nice, easily reconfigurable tabletop accessory

While you can easily find an inexpensive stand for a book or a tablet in the market, it wouldn't come close to the PVC Man. It requires 15 pipes of three different lengths and some endcaps, elbows, and other connectors. Assembling the stand is pretty straightforward, and takes less than a minute once you have assembled and laid out all the components. While you can use a baseball or a light bulb as the head, PVC Man uses a specially assembled head that gives it a robot-like appearance. The head has its own set of components that again can just be pressed into each other. Besides holding a book or a tablet, you can reconfigure it with ease to strike different poses. The intrepid maker behind Specific Love Creations enjoys tinkering with PVC, and has an exhaustive list of PVC projects and life hacks (**hsmag.cc/i11um0**).

CAMERA STABILISER

When Justin was out looking for a stabiliser for his DSLR, he couldn't find one within his budget. So, like any good DIYer, he decided to build himself one with PVC pipes. His hack isn't just cost-effective, but is designed so that it can be held with either or both the hands, and can produce relatively stable images even when used while running, or riding a bike. Like all builds this month, Justin's bill of materials is quite extensive as well. The assembly involves measuring, sawing, hacking, sanding, and gluing the various components, but Justin has explained and photographed the steps in great detail. He suggests you take your time through each step, to make sure you get it right before moving on to the next one. To give you a feel for the contraption, Justin has also included a test video shot using his stabiliser, which is fairly decent.



Project Maker
JUSTIN DERATO

Project Link
hsmag.cc/onMp2A

Left
Based on his research, Justin went with the octagon shape, since it was ergonomic and offered a wide range of filming dexterity

Raspberry Pi PRESS

The MagPi | HackSpace TECHNOLOGY IN YOUR HANDS | CUSTOM PC

3 ISSUES FOR £10

+

FREE BOOK



custompc.co.uk/freebook

Subscribe to The MagPi, HackSpace magazine, or Custom PC. Your first three issues for £10, then our great value rolling subscription afterwards. Includes a free voucher for one of five fantastic books at store.rpipress.cc/collections/latest-bookazines
UK only. Free delivery on everything.

HackSpace
TECHNOLOGY IN YOUR HANDS

FORGE

HACK | MAKE | BUILD | CREATE

Improve your skills, learn something new, or just have fun tinkering – we hope you enjoy these hand-picked projects



PG 78



MAKER MATHS

Some sums to help smooth your builds

PG 80

VINYL CUTTING

Create a personalised pop-up card

PG 86



SCRAP CLOCK

Telling the time with things you can find in a shed

PG 70

SCHOOL OF MAKING

Start your journey to craftsmanship with these essential skills

70 CircuitPython
72 Work holding
76 Printing PETG

PG 90

DESIGN A PLANTER

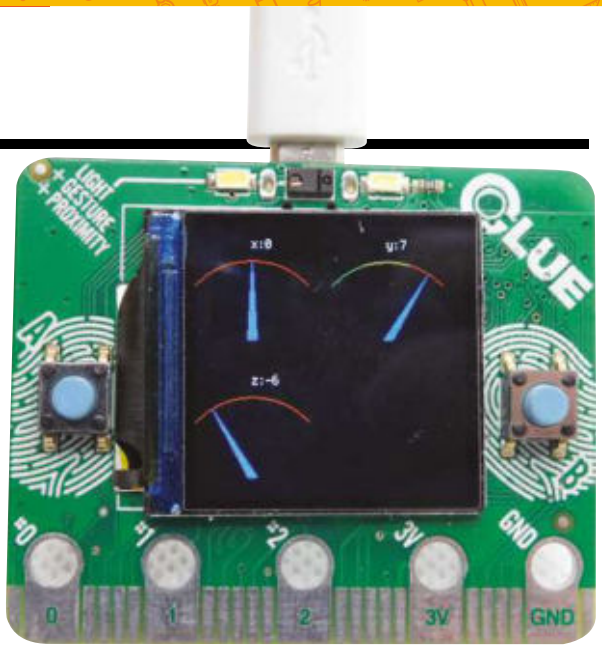
Use Tinkercad to build a new house for your leafy friends

PG 94

CMOS MUSIC

Abusing logic chips for rhythmical sound

Drawing graphs with CircuitPython



Speedometer-style gauges for (almost) any data



Ben Everard

@ben_everard

Ben loves cutting stuff, any stuff. There's no longer a shelf to store these tools on (it's now two shelves), and the door's in danger.

We looked at using displayio to drive graphics on CircuitPython back in issue 22. However, in the past year, the library has seen a lot of development – so let's take another look at how to draw graphics with this microcontroller. Back then, we looked at drawing bitmap graphics on the screen, so now let's take a look at what it's like drawing with primitives like triangles.

We're going to take a look at a simple gauge that displays data a bit like a car speedometer. We've created a library to make this easy – so the basic way of using it is just to copy the **gauge.py** file from **hsmag.cc/1fH3ZZ** to the **lib** folder of your CircuitPython device.

You'll need a screen to make this work. We've tested this out on both an Adafruit CLUE and a Pimoroni Enviro+ FeatherWing. The code we'll look at here takes the accelerometer data from an Adafruit CLUE and displays it as gauges.

```
import board
from adafruit_clue import clue
import displayio
import time
from gauge import Gauge

display = board.DISPLAY

colour_fade=[
    0x00FF00,
    0x00FF00,
    0x00FF00,
    0xFFFF00,
    0xFFFF00,
    0xFFFF00,
    0xFFFF00,
    0xFF0000,
    0xFF0000,
    0xFF0000]

gauge = Gauge(-10,10, 120, 120, value_label="x:")
y_gauge = Gauge(-10,10, 120, 120, value_
label="y:", arc_colour=colour_fade, colour_
fade=True)

y_gauge.x = 120

group = displayio.Group(scale=1)

group.append(gauge)
group.append(y_gauge)

display.show(group)
board.DISPLAY.auto_refresh = True

while True:
    x, y, _ = clue.acceleration
    start = time.monotonic()
    gauge.update(x)
    y_gauge.update(y)
    print(time.monotonic()-start)
```

Gauges are displayio groups, so we can manipulate them as you would other groups, such as repositioning them by setting their x properties as we have done with the `y_gauge` here.

The full code is available on GitHub, but let's take a closer look at gauge here. We add data using the update method which is:

```
    def update(self, val):
        max_angle = 135
        if val<self.min_val: angle = 45
        elif val> self.max_val: angle = max_angle
```

Above
You can fit a lot of data on the CLUE's 240×240 screen

```
        else:
            angle = ((((val-self.min_val)/(self.
max_val-self.min_val)))*(max_angle-45)+45)
        top_point_x = self.mid-int(math.cos(math.
radians(angle))*self.length)
        top_point_y = int(math.sin(math.
radians(angle))*self.length)

        if self.outline: self.arrow =
Triangle(self.pivot1[0],self.height, self.
pivot2[0], self.height, top_point_x, self.height-
top_point_y, fill=self.colour, outline=self.
outline_colour)
        else: self.arrow = Triangle(self.
pivot1[0],self.height, self.pivot2[0], self.
height, top_point_x, self.height-top_point_y,
fill=self.colour)
        super().pop()
        super().append(self.arrow)

      self.data.text = self.value_
label+str(int(val))
```

There's quite a lot of maths here, but basically, a gauge works by having a line (or in this case a triangle) that goes between a pivot point and a point on the arc. The data is represented by the position on the arc the line hits. We calculate this and store it in the `top_point_x` and `top_point_y` variables. With these in place, we can create a new triangle (in the `self.arrow` variable) using the top point, and two corners that stay fixed. These represent the base of the indicator. This is a slightly cheating way of drawing it, and it will mean that the indicator is a little thinner when it leans to either side than it is when it's in the middle, but in practice, this isn't too noticeable.

Since our gauge is a subclass of group, we can use `super().pop` and `super.append` to remove and add items to the group.

As well as the indicator, we also draw an arc that shows the range that the indicating arrow can move in. This can either be one colour or a range of colours that indicate, for example, at what point the value being displayed should alarm the person using it.

At the time of writing, the displayio library doesn't include the ability to draw lines, so we have to cheat a little and build our arc up out of triangles. If two corners of a triangle are the same, then we get a line. This isn't the most efficient way of doing it, but it works.

All this is done in the `draw_arc` method, which returns a list of triangles that can be added to our group.

```
def draw_arc(centerpoint_x, centerpoint_y,
length, start_x, end_x, num_sections, colour,
height, colour_fade=False):
    triangles = []
    lastpoint = [start_x, int(math.
sqrt(length*length-(centerpoint_x-
start_x)*(centerpoint_x-start_x)))]
    increment = (end_x - start_x) / num_sections
    counter = 0

    for i in range(0,num_sections):
        if colour_fade: this_
colour=colour[counter]
        else: this_colour = colour
        next_x = start_x + (i+1)*increment
        nextpoint = [int(next_x), int(math.
sqrt(length*length -
(centerpoint_x-next_x)*(centerpoint_x-next_x) ))]
        triangles.append(Triangle(lastpoint[0],
height-lastpoint[1], lastpoint[0], height-
lastpoint[1], nextpoint[0],height-nextpoint[1],
outline=this_colour))
        lastpoint = nextpoint
        counter = counter+1

    return triangles
```

There's a bit more code, but it only links everything together. Take a look at the GitHub project to see everything.

The speed our gauge runs at depends a lot on the size of it. We found that on a CLUE, we could get about 7fps with two gauges each 120 pixels across, which is plenty for most purposes (if your data's displaying faster than this, a gauge may not be the most appropriate display anyway). ◻

Below
The gauge library is more adaptable than we've shown here. Take a look at the source code for all the options

```
board
adafruit_clue import clue
displayio
time
auge import Gauge

y = board.DISPLAY

_fade=[
00FF00,
00FF00,
00FF00,
FFFF00,
FFFF00,
FFFF00,
FFFF00,
FF0000,
FF0000,
FF0000]
= Gauge(-10,10, 100, 100, value_label="x:")
e = Gauge(-10,10, 100, 100, value_label="y:", arc_colour=colour_fade, colour_fade=True)
e = Gauge(-10,10, 100, 100, value_label="z:")
e.y=120
e.x = 120


= displayio.Group(scale=1)
```

Making bench blocks!

Bench blocks are an incredibly useful tool in a workshop. Let's make some from a variety of materials

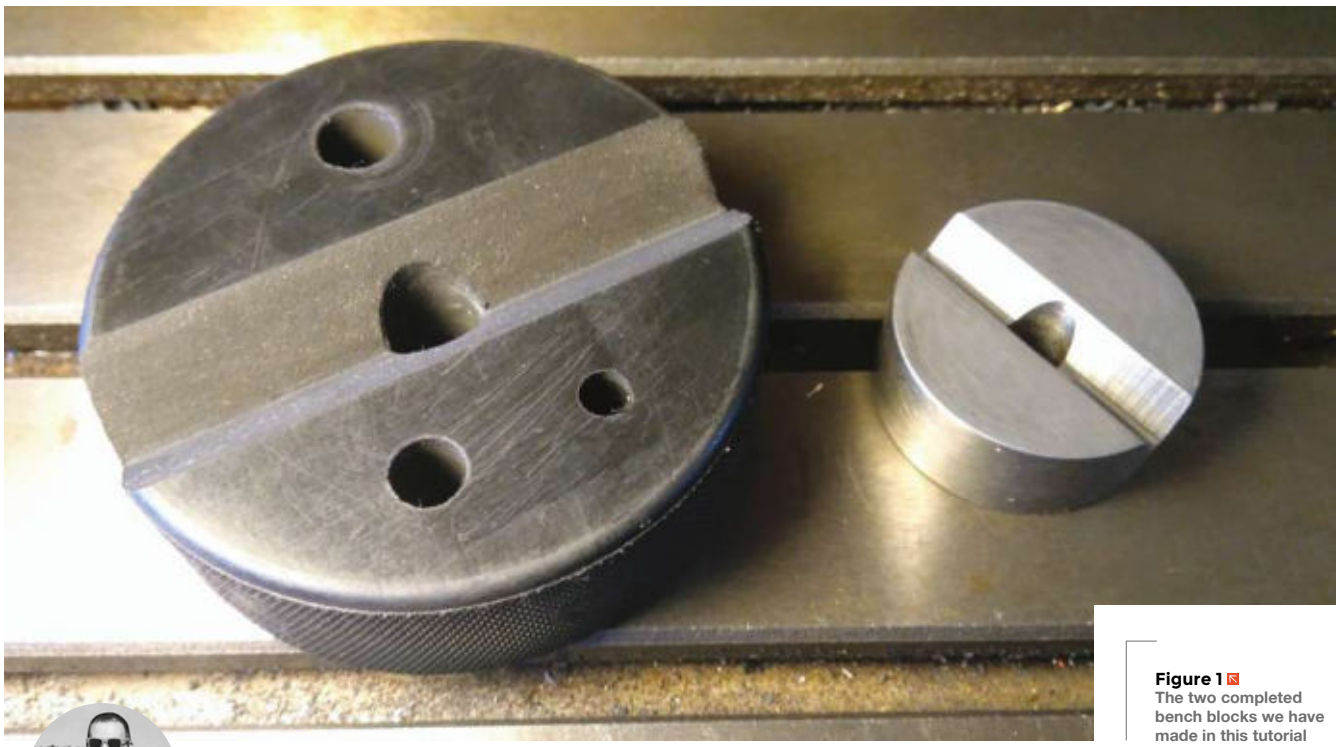


Figure 1 The two completed bench blocks we have made in this tutorial



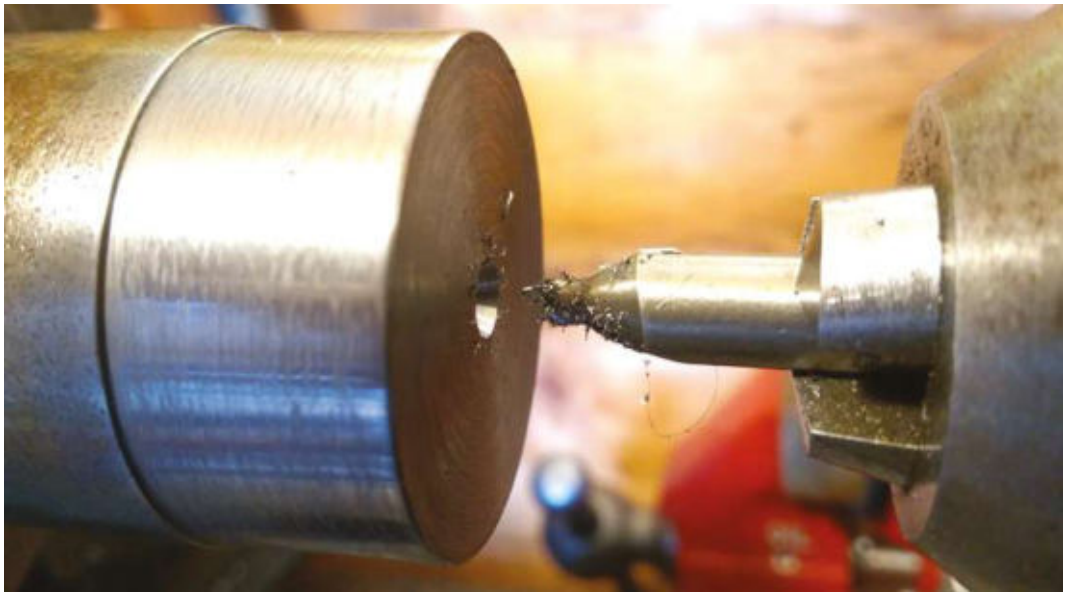
Jo Hinchliffe

@concreted0g

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY space. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open source. He also has a shed full of lathes and milling machines and CNC kit!

Bench blocks are commonly available tools that are commonplace in numerous trades, such as gunsmithing and jewellery making. They are usually circular and have a vee groove cut across them, with a hole through the centre. They are useful for holding cylindrical workpieces for a variety of tasks, such as drilling holes through round bars, pushing pins and other items through round workpieces, and much more. They can also be used as small anvils for punching, marking, and stamping work, and many of them come with extra holes through them that can be used as drill guides, or more commonly tapping guides, helping a tap to remain vertical to the workpiece.

Bench blocks are reasonably affordable to buy, so you may question why it's worth making them! The two bench blocks we are making in this article both differ slightly in design from the commonly available ones, and have alternate uses. The first one we have made is much smaller and has a different angle of groove than the commercial ones, enabling it to hold hexagonal material firmly. Most commercial bench blocks are made from hardened steel and finished to a high standard but, being hardened steel, they are usually harder than the workpiece, and this can lead to them marking workpieces. The second bench block we will make in this tutorial is therefore machined from a hard rubber hockey puck, which is a perfect material for a bench block that won't mark the surface of a piece of work.



YOU'LL NEED

- Milling machine
- Lathe
- Hockey puck
- Piece of mild steel
- 60-degree cutter
- End mill
- Drill bits

For both of these bench blocks, the dimensions and accuracy aren't too critical, so in this tutorial, we aren't particularly giving exact measurements or plans but, rather, urge you to experiment with a design that you feel might be useful. For the smaller metal one, we used a piece of EN1A mild steel, around 35 mm in diameter, and mounted it up in the three-jaw chuck on the lathe. The first operation was to face the stock to create a smooth, flat surface, and we turned a small amount off the diameter to make it concentric with the lathe, but also to clean up the outer surface (**Figure 2**). First, using a centre drill to start the hole accurately, we used the lathe tail-stock to drill the workpiece to a depth of around 30 mm, stepping up in drill size to a final size of 8 mm.

We removed the workpiece from the lathe and roughly cut the work to length (around 20 mm), plus a couple of millimetres using our horizontal band-saw. We re-chucked the workpiece with the cut end facing away from the chuck, and then faced the other edge to make it flat and parallel to the opposite side.

We removed the work from the lathe and de-burred it, before setting up on the milling machine to continue. We clamped the workpiece directly to the worktable on the lathe, and ensured it was clamped securely (**Figure 3**). For this smaller bench block, we wanted to cut a 60-degree groove into it, as we had the idea that the faces of the vee would match small hexagonal workpieces enabling these to be held securely, and we could also use it to then index hexagonal work. To cut this vee we had a carbide cutter with a 60-degree cutting geometry. We placed the cutter into a suitable collet holder in the spindle and then used the hole we had drilled to help centre the cutter to the workpiece. This is not the most highly precise way of doing this but, again, the tolerances on this bench block are not critical. →

TAP TAP

We added numerous extra holes to the hockey puck rubber bench block that can be used in a variety of ways. One main idea for them is they can hold another tool vertical while placing a workpiece underneath the bench block. These tools could include round stamps, centre punches, or even roll pin punches. In the image, we have set up an M8 tap and tap holders in the 8 mm hole. It's really helpful to be able to keep a tap vertical, especially when starting to tap a hole.





QUICK TIP

We covered facing and drilling operations in the lathe in issues 15 and 20.

Figure 2 Cleaning up a piece of stock by facing and turning, and then centre drilling the hole

Figure 3 With the item well clamped, we began to use the 60-degree cutter to create the groove



Figure 4
Drilling the hockey puck central 10mm hole was the first time we had cut rubber. Note the large amount of dust it produces!

GROOVY BABY

Having centred the tool to the workpiece, we then locked the X axis and began to use the Y axis travel to cut the vee slot. When using tools which change diameter over their length, it can be difficult to calculate the desired rpm, and so we experimented setting the tool to around 500 rpm, and this seemed to give satisfactory results. We cut the vee with a succession of passes, making the later passes less deep as there is increased load on the tool due to the increasing contact area as the vee slot forms. We added more cutting fluid for the later cuts and decreased our feed rate.

> Having drilled the puck, we then began to set **up an adjustable angle plate on the milling machine**

Having de-burred this small bench block, we are pleased with how it has turned out (**Figure 1**). It holds both round stock and hexagonal stock very well. We found this smaller size of bench block is particularly useful employed as a kind of a fingerplate to hold work in hand when, for example, using a bench grinder.

Next, we turned our attention to making a bench block from the hard rubber hockey puck. We had seen that people had done this on various forums, and were impressed with the solidity of the rubber puck, which definitely won't mark most metals. We had never machined rubber before, and we also wanted to make this bench block to have the more traditional 90-degree vee slot, which means that it can hold round stock but also square stock on its corner.



Figure 5
Setting up the adjustable angle plate using an engineer's square, so we can use a regular end mill to cut a 90-degree groove

We began by marking the centre of the workpiece and drilling a 10mm hole through (**Figure 4**). The rubber cuts extremely freely, with little resistance, so do take care not to plunge straight through the work! Having drilled the puck, we then began to set up an adjustable angle plate on the milling machine to be able to use a regular end mill to cut the 90-degree vee groove. Using an engineer's square to ensure that the adjustable plate is square to the machine ways guaranteed that we would get a straight cut once the tool was set in the correct position (**Figure 5**).

LET'S GET OUR GROOVE ON (AGAIN)!

Having set up the angle plate, we began to clamp the puck to the plate. It's quite difficult to clamp, as the rubber compresses under the clamp and begins to distort the puck out of shape. Conversely, the rubber requires very little clamping force to stay in position. We ended up with a makeshift clamp on the lower part of the puck and a small toolmaker's clamp on the upper part, which doesn't look like much clamping surface but held the workpiece securely while still leaving room for the tool to be positioned (**Figure 6**).

Centring the spindle to a workpiece that is clamped at 45 degrees is a challenge if you want to be very accurate. With a bit of trigonometry and using some dial gauges, it certainly is possible, but this seemed like overkill for an item that doesn't really need huge accuracy. It is very difficult to see, but we marked the centre line of our puck in pencil and continued this mark vertically down the side of the leading edge of

Figure 6
Cutting the groove with an end mill with the hockey puck creatively clamped to the angle plate



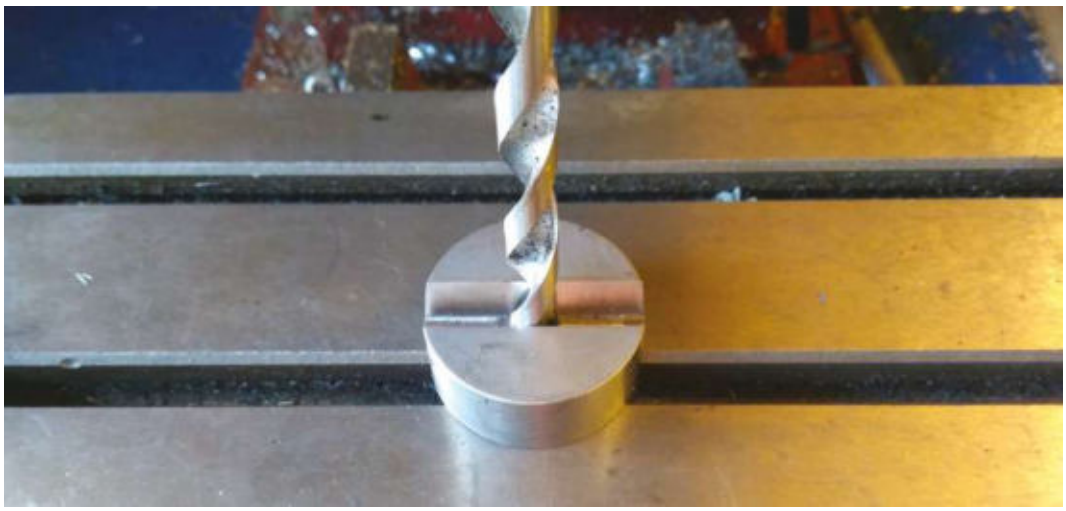




Figure 7
A quick way to align the centre hole of the bench block to the spindle of the milling machine is to use the same diameter drill bit and place it into the hole

Figure 8
A small brass bar cross-drilled with a 3mm hole accurately placed on the centre line

the puck. When clamping the puck to the angle plate, we used a very thin drill bit in the spindle, as a crude but effective reference point, making sure that the drill bit tip was on the centre line marked on the puck at either end of the line.

Next, we fitted the end mill, which was 16mm in diameter. We moved the spindle over by 8mm to then ensure that the edge of the tool was aligned to the centre line of the puck. We moved the end mill down to the puck, and touched on and zeroed the dials on the milling machine handles.

To cut the vee, we then repeatedly moved the end mill down and the X axis across by the same amount. We began with a 2mm movement down and across. While this might represent a large cut in a hard material, it was a trivial amount for the rubber our puck was made from. Having not machined hard rubber before, we did some guess-work on the cutting speed and feed rate, but indeed it is fair to say that it is extremely free cutting and can be worked at high speed. However, it's possible to work it too fast so that the rubber begins to melt, so some experimentation is needed.

Continuing in larger steps, we cut the 90-degree vee until it was around 10mm deep. Removing the work from the angle plate, we decided to add some holes to this bench block that could act as a guide for hand tapping threads. We added a 6, 8, and 10mm hole to the side of the block just using common HSS drill bits. In the future, we may add holes to the smaller steel bench block, but for those, we may drill them slightly under-size and ream them to accurate dimensions.

Having recovered from the smell and tidied up the rubber chips, we set about putting our bench blocks to the test. The rubber bench block is excellent in that the rubber creates a slight grip on a workpiece, and it is very easy to hold work in it securely. Another advantage of the vee groove geometry is that you can use them on the milling machine, or a drill-press, to set a drill bit on the centre line of the vee. Simply place a drill bit into the chuck and then bring the spindle down to place the drill in the groove, and the bench block will settle with the drill touching each side of the vee. This makes setting up to drill through a workpiece quicker. Another option we tried is we used the smaller bench block and centred it to the drilling spindle by placing an 8mm drill bit into the chuck and aligning it in the centre hole of the block (**Figure 7**). We then very carefully, without moving the block, placed a small brass bar in the groove and clamped it all down. We centre drilled and then drilled a 3mm hole through the brass bar, knowing that the drill spindle was centred on the centre point of the workpiece. This works extremely well and makes a challenging task pretty trivial to complete (**Figure 8**).

Finally, while our smaller steel bench block is too small, we wanted to see if the hockey puck bench block could be used as a small anvil. It's not the most efficient anvil, but the hard rubber does well in that it stays in position on a workbench due to rubber's inherent grip, and it also absorbed the energy of a blow well and didn't bounce on the workbench when struck.

To conclude, we enjoyed the challenge of making these, and recommend picking up a cheap hockey puck and making one yourself. A useful tool for a variety of tasks.

QUICK TIP

We found that the rubber puck created chips that were quite dusty when machined, so a face mask is recommended. It's also quite smelly when worked!

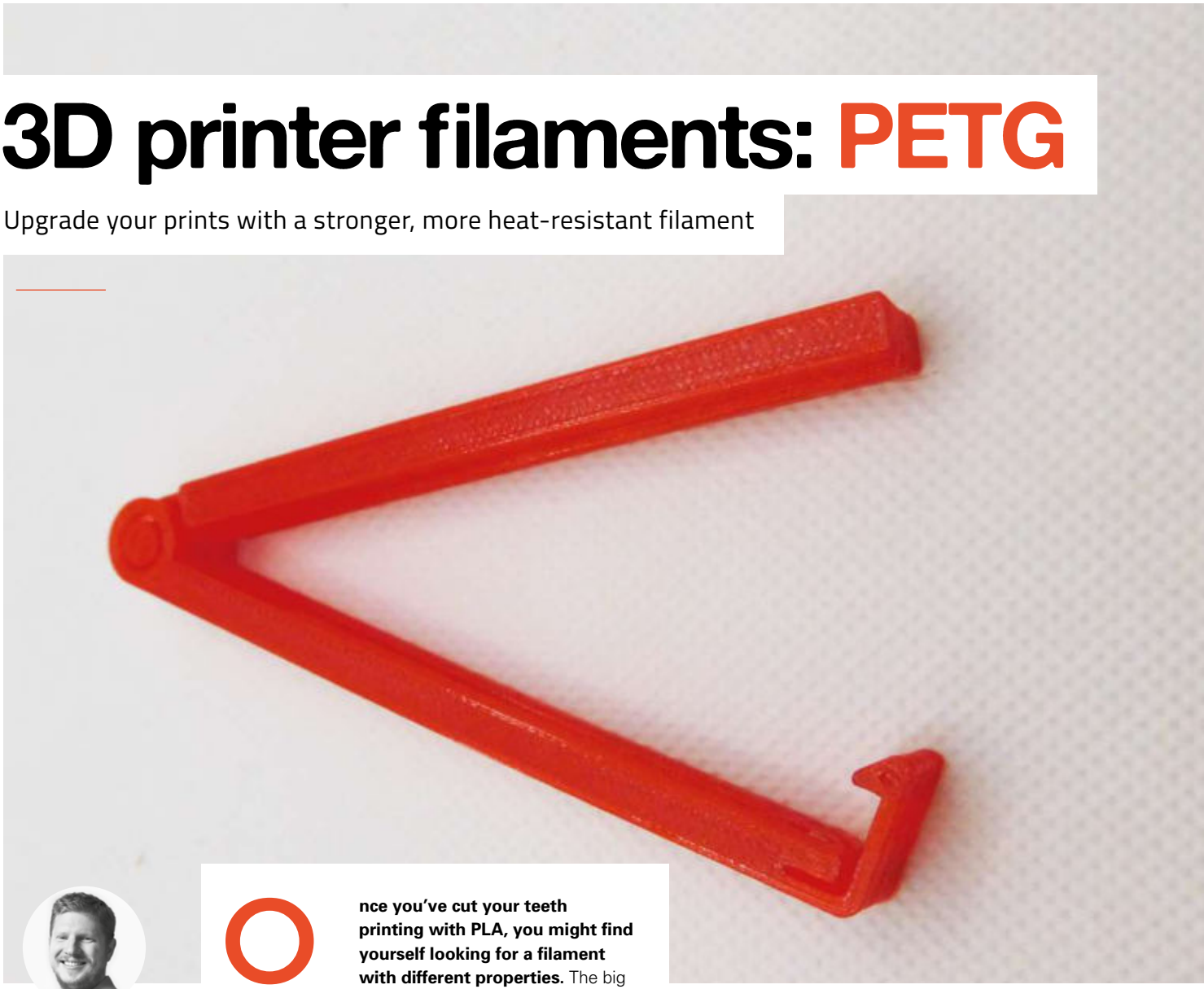
HOLDING HEX

We tested our idea of the smaller bench block being capable of holding hexagonal workpieces and found that it was excellent at it. Placing a hexagonal piece into it means that the uppermost face of the hexagonal piece is held flat and parallel with the bench block surface. We realised that this could, in turn, be used to create a hexagonal piece from a piece of round bar by milling a flat on the top of the bar, and then indexing that flat round the vee cut and creating the other flat surfaces.



3D printer filaments: PETG

Upgrade your prints with a stronger, more heat-resistant filament



Ben Everard

@ben_everard

Ben loves cutting stuff, any stuff. There's no longer a shelf to store these tools on (it's now two shelves), and the door's in danger.

Above
Mechanical parts, particularly those that need a little flex (such as this clip), are well-suited to PETG

Once you've cut your teeth printing with PLA, you might find yourself looking for a filament with different properties. The big downsides to PLA are that it's brittle, not stable at temperature, and not particularly strong. There's one filament that improves on all of these while still being cheap, easily available, and relatively easy to work with: glycol-modified polyethylene terephthalate (more commonly known as PETG).

Let's take a look at what it takes to make the switch to this filament. The very first thing that you need to make sure of is that your printer can cope with PETG. Most modern printers can, but it is a little more demanding than PLA. Firstly, you'll need the hot end to be able to reach a higher temperature – at least 220°C (we print at 240°C). This can cause a particular problem for printers without all-metal hot ends. Bear in mind that just because a hot end can reach a temperature, it doesn't mean it should. The most common damage caused by heat is a burned PTFE tube. Check your printer's documentation for guidance on maximum temperatures.

Secondly, you'll need a heated bed that can get up to 70°C. If you can meet these two requirements, and you can get PETG in a format that fits your printer (usually 1.75 or 2.85mm filament), then you should be able to print PETG.

You need to prepare your print bed a little differently for printing PETG to ensure two things. Firstly, that you can get the filament to stick, and secondly, that you can get the final print off without damaging it. PETG sticks much more firmly than PLA, so don't underestimate this.





Above
Many people have come to PETG in the last couple of months because it's a good choice for medical equipment, such as these face shield parts. It's more sturdy and easier to sterilise than PLA

Left
As PETG is often slightly transparent, you may be able to see in the infill

The approach differs a little between bed types. For flexible beds (such as the magnetic beds on recent Prusa printers), removing the print is a little easier, so you may not need any special release agent on the bed. However, do check what bed preparation is recommended. For example, on Prusa PEI beds, it's advised that you use the textured bed and clean with Windex or IPA (never acetone).

If you do anything to change the height of the bed (such as using a textured sheet or adding painter's tape), don't forget to adjust the Z axis accordingly either through relevelling or in firmware.

For glass beds or other ridged beds, you'll need to think a little more about getting prints off. This usually means putting something on the bed, commonly blue painter's tape or an adhesive like Magigoo. It's worth experimenting with a few different options to see what works best for your setup.

SOFTWARE SETTINGS

Your slicer should have a PETG setting, but it can be worth taking a bit of time to make sure you've got the settings as good as can be. A temperature tower will print a vertical structure at different temperatures so you can see how overhangs, bridges, and stringing are affected by temperature (there are several options available on Thingiverse).

PETG is more prone to stringing and blobbing than PLA. This can be solved either in print settings or post-processing. In print settings, you'll need to tweak the temperature and retraction settings. Small stringing tests are available on most 3D model websites, and these should help you dial in your settings. In post-processing, you can use a heat gun, flush cutters, or scrape a blade across the surface to remove these artefacts (though be careful if using a blade).

You may find that your hot end has a more difficult time holding a stable temperature with PETG than with PLA, as the required temperatures are higher. If you notice fluctuations, slow the print down a little. Less plastic flowing through the extruder will make it easier for it to hold the temperature.

As with many things in 3D printing, there aren't any absolutes, and printers all behave a little differently, so the best thing is to experiment with settings until you find ones that work for you. Happy printing!

WHEN IS PETG APPROPRIATE?

No printer filament is perfect, and often we're not looking for one with perfect properties, but the most applicable properties to our application. Let's take a look at the good and bad aspects of PETG:

The good

- More impact resistant than PLA.
- Slightly more flexible than PLA.
- More temperature resistant than PLA.
- More forgiving than ABS (and most other higher strength filaments) when printing.
- The translucent colour can give a nice effect.

The bad

- At the time of writing, global supply is low due to its use in medical applications. Consider using a different filament until supply improves, unless it's essential.
- Unlike PLA, it's not biodegradable.
- A little less forgiving than PLA.
- Can be difficult to remove from the heat bed.
- Not all printers can achieve the necessary temperatures.

TUTORIAL

Maker maths

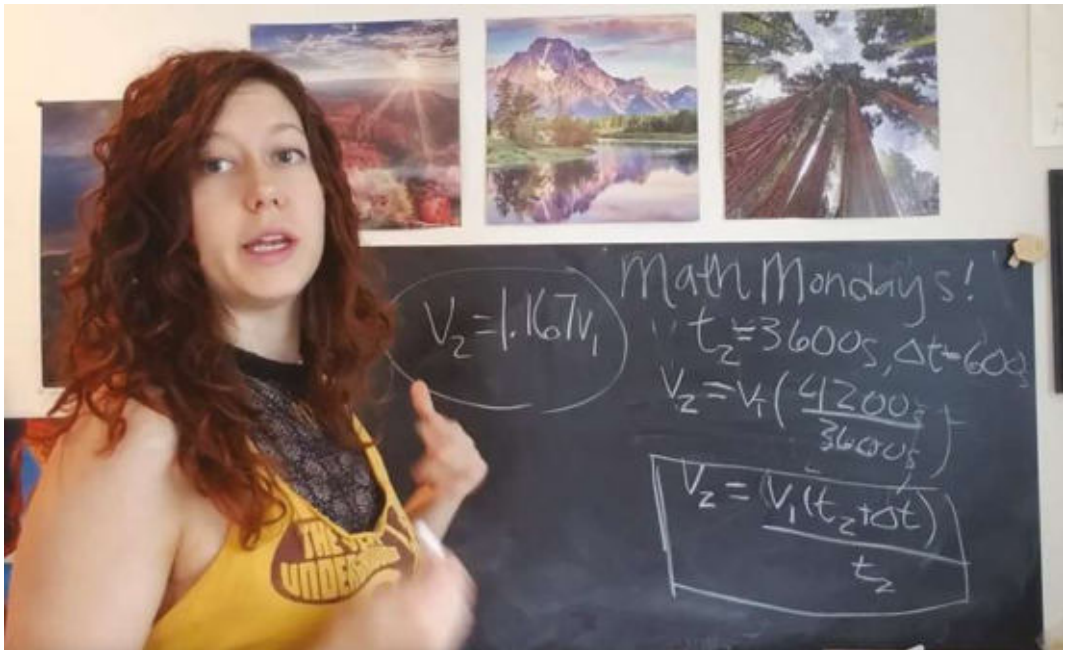
Make numbers work for you and your projects



Gareth Branwyn

@garethb2

Gareth has been a lifelong practitioner (and chronicler) of DIY tech, media, and culture. He is the author of ten books, including *Tips and Tales from the Workshop*, and is a former editor for Boing Boing and Wired.



Above
Mathnaut Jen Foxbot has an excellent video on YouTube showing what to do if you have two 3D printers going and you want to speed up the second printer so that both prints finish at the same time. Take a look at hsmag.cc/M4NQUd

The ultimate maker maths – when considering making something over buying it, you want to be realistic in terms of calculating the cost and the time it will take you. If a widget you're thinking about buying costs $1500, and you calculate that the realistic cost of making it yourself is $1300, is it worth doing? How you calculate this depends a lot on you, how you value your time, if you think the learning you'll do in the process is worth it, etc. In some situations, even if it costs more to do it yourself, it still might be worth it for you. And then there's the ultimate X factor: the sheer joy of making, and ultimately using, an object or tool that you made yourself. That calculation is often the hardest to make.

Estimating project time Carefully, figure out how long a project is realistically going to take and then double it. Especially if it's a job for someone else, you will look like a miracle-worker when you deliver it ahead of schedule. It's also great for your own sense of accomplishment to finish projects ahead of time.

Estimating project costs Come up with a realistic figure for how much the supplies and materials for a project will cost and then triple it. Besides probably needing more than you think, there are also all of the incidental costs that you don't or cannot foresee.

Adding a hassle tax If you're working on a project or with a client that you fear might be more difficult than usual, add a hassle tax to your job estimate. This is the extra time, hassle, and heartache this situation is likely to cost you. The amount of tax may vary. When this author ran a graphic design business many years ago, the hassle tax was 20%. If the client baulked at the price, so be it. If they were willing to pay it, the extra hassle was paid for.

Calculating safety factor When working with anything that will come under a live load – something

that will come under stress – take the stated load rating of the materials you are working with and multiply by five (or more). Static loads are obviously less prone to stress failure than live loads, but you should always err on the side of caution. In some engineering schools, students are taught to calculate load safety and then told to add a zero to the result.

Calculating hourly rates based on probability From Miguel Valenzuela (inventor of the PancakeBot) comes this gem: one of the biggest challenges we have as freelancers is calculating how much our hourly rate should be. Using probability helps in calculating what you should charge based on the probability of work. Here's an example:

Shelly wants to make $100,000 per year, and has the ability to work 2000 hours per year. If she works 100% of that time, then she will need to charge $50 per hour. The problem is that Shelly doesn't actually work 100% of the time. She has calculated that the probability of her working is actually 62%. So, the minimum hourly rate she must charge is:

$$\frac{\$\ 100,000}{Hours \cdot Probability\ of\ Hourly\ Work} => \frac{\$\ 100,000}{2000 \cdot (.62)} = \frac{\$\ 80.6}{hour}$$

Now remember, Shelly can work more or less than 62% of the time, so her probability of work is merely a planning and forecasting tool; it is not set in stone.

Estimating the square roots of small numbers When figuring out the square root of a small number, it's helpful to know that the square root of 1/1000 = ~.031.

Converting decimals to fractions To convert from decimal to fraction, multiply the amount to the right of the decimal point by the desired denominator to get the numerator. For example, 0.5 ×16 = 8, so 0.5 is 8/16.

Converting model scales and real-world scales to model sizes If you work with model scales, such as those used in model railroading or scale modelling, you sometimes need to convert between the designated scale sizes (e.g. HO-scale or 1:87) and real-world measurements. To do this, there are a number of scale conversion calculators online. Here is one: **hsmag.cc/2cONIK**.



Above
1-2-3 blocks can make it easy to mark out common sizes

Enter mathematical equations rather than calculating numbers to enter into modelling software Most modelling programs will allow you to enter equations into their parameter fields so that you don't have to actually calculate the number first.

Handling very large or small numbers in your head Multiplying or dividing large or small numbers is easier if you shift the decimal places on both numbers until you have numbers you can deal with. If the two numbers are on the same side of the equation, shift one to the right and the other to the left, like so:

X = .002 * 10,000
X = .02 * 1,000
X = .2 * 100
X = 2 * 10
X = 20

If they're on opposite sides of the equation, shift them both in the same direction by multiplying or dividing both by 10:

100X = .5
10X = .05
X = .005 □

THANKS
Bryce Lynch, Michael Colombo, Ross Hershberger, Dave Porter, Becky Stern, Jen Foxbot, Miguel Valenzuela

TUTORIAL

Pop-up cards

Automatically cut, score, and draw a pop-up card using a vinyl cutter



Poppy Mosbacher

@PoppyMosbacher

Poppy loves getting tech into the hands of people who do traditional crafts. She is helping set up a makerspace in Devon, and was a director of Build Brighton makerspace.
poppymosbacher.com

Personalised pop-up cards are a playful way to keep in touch with absent friends. And vinyl cutters are the ideal tool for making them, as they can cut and score light card ready for folding. Most machines also have a pen holder to draw colourful details and mimic handwriting. We've used a Cricut Maker because it's a popular budget machine, but the method can be adapted for other vinyl cutters, or made by hand using a craft knife or scissors.

All pop-up cards are based on simple folding techniques, such as the box mechanism, which we're using to support a sailing boat. Once you learn the method, it can easily be used for a wide range of shapes, such as a pile of presents or famous landmarks.

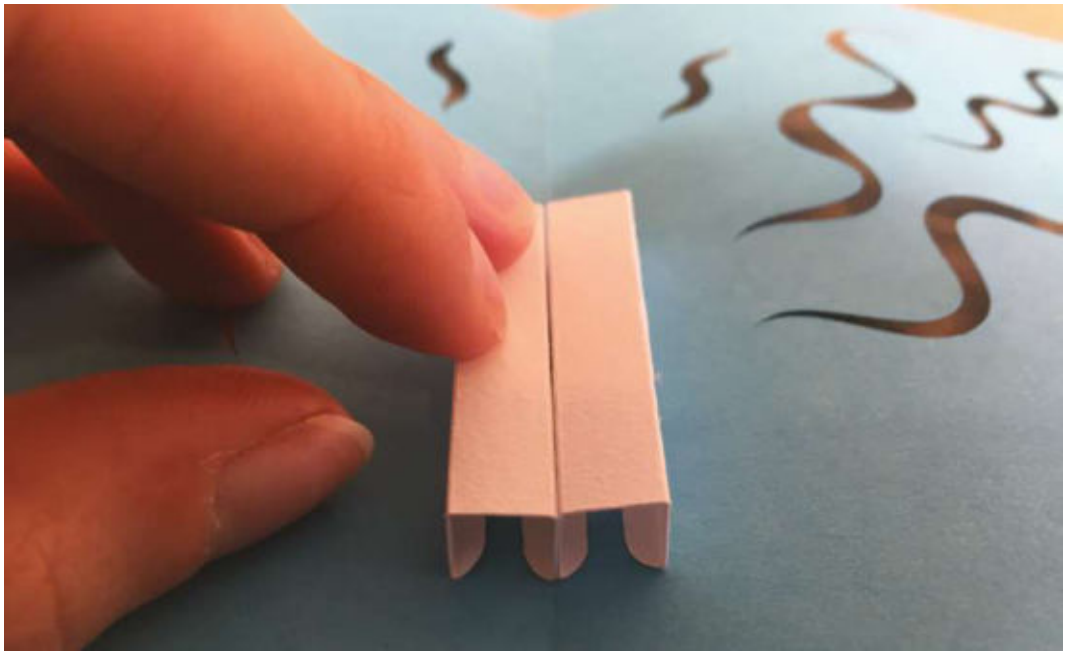
PUSH THE BOAT OUT

The first step is to draw all the pop-up parts in a vector graphics software. We're using Inkscape, because it's free and open-source.

Draw individual shapes for a mast, sails, and rigging using the Pen and Rectangle tools. Position the shapes so they resemble a sailing boat and add a rectangle at the base, which will be glued inside the box mechanism. Make sure all the shapes slightly overlap each other, and then move the back sail to the edge of the screen because it will be cut separately from coloured paper. Select all the other shapes and go to Path > Union to turn the individual shapes into a continuous cut line.



Right
Add an element of surprise to homemade greetings cards



Left
Position the box mechanism across the centrefold

Create the first half of the box mechanism by drawing a portrait-orientated rectangle and use the Node tool to round the corners. Split the rectangle into five segments using four vertical lines which will be scored by the vinyl cutter and then folded. The outer segments are for glue, and the inner segments will form the height and width of the box. Copy and paste the rectangle and vertical lines to complete the box mechanism.

Draw the sides of the hull, with one of the sides having a glue tab at the front. Add a decorative, horizontal line along each side. If the sides are both pointing in the same direction, flip one of them by going to Object > Flip Horizontal, so the decorative lines will be visible when the hull is assembled.

> Make sure all the shapes slightly overlap each other **and then move the back sail to the edge of the screen**

Create the back of the hull by drawing a landscape-orientated rectangle with rounded corners. Add a vertical fold line in the centre of the rectangle and on both ends to form glue tabs. Set the width of the rectangle (not including the glue tabs) to match the width of the box mechanism, and the height to match the sides of the hull.

MAKE WAVES

Create the ocean background by drawing a rectangle and add a central line for the fold. Click on the Pencil tool and from the toolbar that appears, set the shape to Ellipse. Also click 'LPE Based Interactive Simplify', which allows you to edit the smoothness of lines after you've drawn them. Draw wavy lines to represent the wake of the boat in the water, then select the lines and go to Path > Path Effects and under Pattern along Path to choose a width for the waves. Save the design as an SVG file.

Open a new project in Cricut Design Space, or adapt these instructions if using other software. Click the Upload button, then click Upload Image. Drag and drop the SVG file and follow the prompts to add the file to the project. Sometimes the scale changes when designs are imported, so if this happens see the Changing Scales box (overleaf) for how to resolve it.

Select all the fold lines, and use the Line Type drop-down menu to set them to Score. You may need to ungroup objects to select individual lines or, if lines are hidden, right-click on an object and click Send to Back. →

VINYL CUTTERS ARE VERSATILE MACHINES

As well as cutting vinyl for T-shirt designs and signage, these machines can cut paper, cardboard, and stickers. Some also cut fabric, leather, and softwood.

Professor Neil Gershenfeld, the original founder of Fab Lab, says vinyl cutters are one of the best, but under-used, tools in makerspaces, They are usually cheaper than laser cutters and take up less space, yet can do some of the same tasks – without leaving burn marks.

QUICK TIP

A spatula can be useful for removing delicate pieces, such as rigging, from the cutting mat.

YOU'LL NEED

- **Cricut Maker®,** or other Vinyl Cutter
- **Vinyl cutter grip mat**
- **Scoring stylus**
- **Vinyl cutter compatible pens**
- **Computer**
- **Inkscape software** (free)
- **Cricut Design Space** or similar vinyl cutter Software
- **White card** (160gsm)
- **Coloured paper** (Blue and red)
- **Glue** (e.g. UHU® All-Purpose Glue)

Right
Cutting shapes from the front of the card allows the blue backing paper to show through

Below
Thin rectangles, designed to look like rigging, provide structural support for the mast



Use the small square next to the Line Type drop-down, to assign colours to the cut lines. It doesn't matter which colours you choose, as long as each material is given a separate colour. To follow this tutorial, assign white to the mast section, grey for the hull and box mechanism, red to the back sail, and blue to the ocean background.

ADD A PERSONAL TOUCH

Click on the Text button and type words into the text box that appears. From the text toolbar, you can change the font, size, and curve effect. Set the Line Type to Draw and assign colours to match your pens.

Although all the parts will look like they are in the right positions, there is one last step before you click Make It. Individually select and attach all the lines and text to the main shapes, otherwise they could move to random places.

Then click Make It. The shapes will be split onto virtual cutting mats, based on the cut line colours. Set the material size using the drop-down menus. Select the white mat and place white card on the real cutting mat, matching the position shown on screen. Then click Continue.

Connect to the vinyl cutter using a lead or Bluetooth connection.

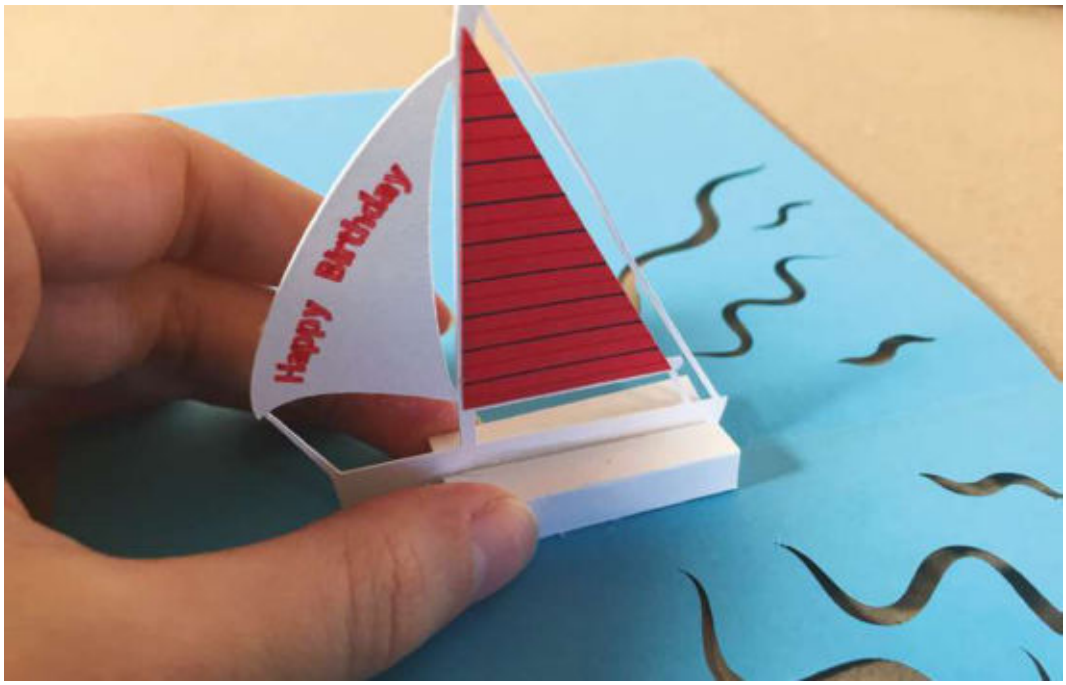
Click Browse All Materials and choose Light Cardstock. Then click Edit Tools and select the scoring stylus. Load the cutting mat into the vinyl cutter and place the scoring stylus into tool holder 'A' and the fine-point blade into holder 'B'. Press the flashing Go button on the machine to start.

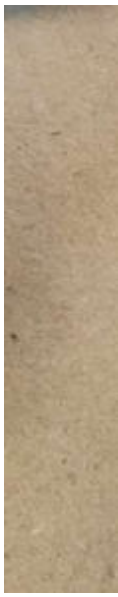
AUTOMATIC WRITING

The Cricut will pause after it scores the fold lines, for you to swap the scoring stylus for a pen. Click the Go button to resume. If your design includes multiple colours, the machine will pause for each pen before moving on to cutting.

To prevent the card from curling, peel the mat away from the pieces of card, rather than peeling the card away from the mat.

Select the next virtual cutting mat and set the material to Copy Paper, then repeat the process for the other shapes on red, white, and blue paper.





QUICK TIP

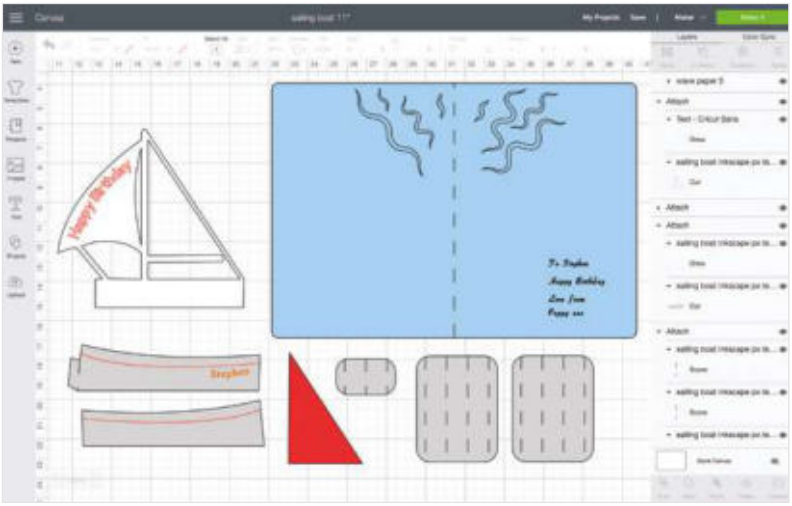
Vinyl cutters can be used for making envelopes and automatically write the address on for you.

CHANGING **SCALES**

The software that comes with the Cricut is great for basic shapes, but anything more complex is probably easier to draw using a different vector graphics software, such as Inkscape.

If you find the scale changes when transferring designs between software, the Inkscape website shares a simple technique for how to resolve it. Go to File > New from Template > Default px. Then copy and paste your design into this template. Save the file and upload to Cricut Design Space, or your vinyl cutter software as usual.

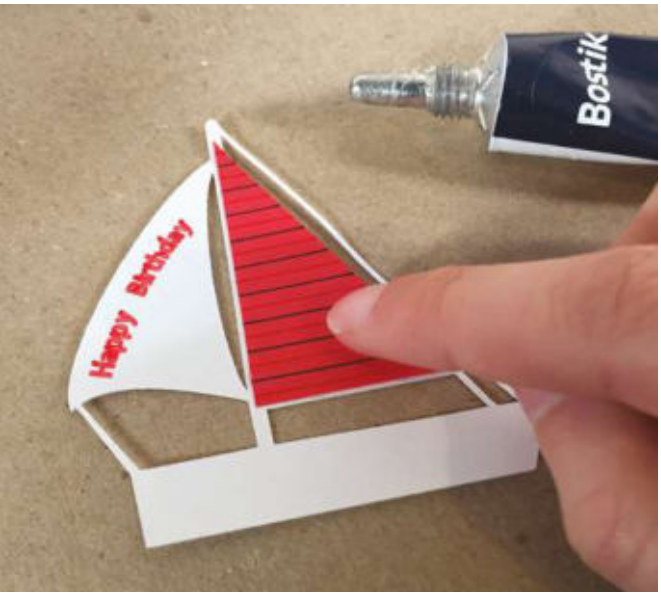
For more information go to: **hsmag.cc/V2JhwX**.



MAKE IT POP

Fold the paper pieces along the score lines, folding forwards and backwards. Apply glue to the mast and attach the back sail.

Glue along one tab of one of the box mechanism pieces. Attach it along the centrefold of the ocean background, so that the glued tab is on one side of the fold and the rest of the box piece lies flat on the other side of the fold. This sets the position for the back of the boat. Fold over a third of the box piece plus the second tab, and carefully apply glue to this tab. Close the ocean background, keeping the box piece in position and press down to make the glue stick. When you reopen the background, the first part of the box mechanism should be glued in place. Use the same method to align the second box piece on the other side of the fold.



Flatten the box pieces and apply glue to the side of each box that will be like a sandwich in the middle of the boat. Stand the mast section along the centrefold with the back of the boat lining up with the back of the box mechanism. Gently push the boxes into shape.

Apply glue to both tabs on the back section of the hull, and stick it to the back end of the box mechanism.

Attach the hull sides together using the glue tab. Then apply glue to the sides of the box mechanism and attach the hull.

Gently squeeze the back of the boat until the back piece starts to fold outwards, then bring one side of the ocean background over, so the pop-up folds flat. Reopen the ocean background to check the pop-up works.

Fold a piece of white card in half and decorate the front. We continued the nautical theme by using the vinyl cutter to cut an anchor emblem from **freepik.com** (**hsmag.cc/l4LXZs**). Hide the pop-up inside by attaching the ocean background to the card with glue.

USEFUL RESOURCES

For inspiration and techniques to make more pop-up designs, check out Duncan Birmingham's Pop-Up Channel: **hsmag.cc/gk06CP**.

Detailed instructions on how to use the pen tool, including how to convert straight lines to curves with the Node Tool, can be found at: **hsmag.cc/H7ChYy**.

For tips on using the pencil tool go to: **hsmag.cc/7PMWYm**.

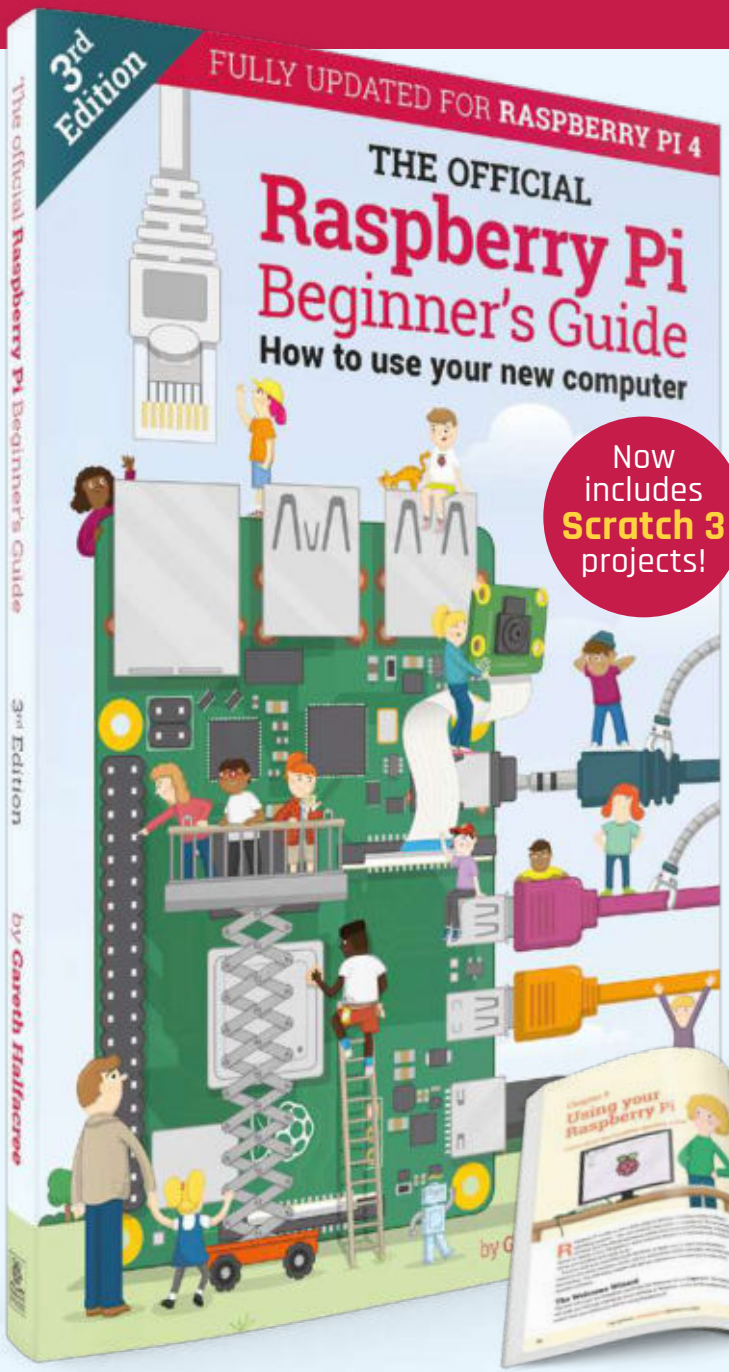
QUICK TIP

Measure the thickness and weight before posting, as it may need a large letter stamp.

Above
Ready to start cutting!

Left
Decorate the sails with text, lines, or shapes

THE OFFICIAL Raspberry Pi Beginner's Guide



The only guide you need to get started with Raspberry Pi

Inside:

- Learn how to set up your Raspberry Pi, install an operating system, and start using it
- Follow step-by-step guides to code your own animations and games, using both the Scratch 3 and Python languages
- Create amazing projects by connecting electronic components to Raspberry Pi's GPIO pins

Plus much, much more!

£10 with FREE worldwide delivery



Buy online: magpi.cc/BGbook

BOOK OF MAKING

VOLUME 2



ONLY £10

WHSmith

BARNES&NOBLE

Available on the App Store

Google Play

THE BEST PROJECTS FROM

HACKSPACE MAGAZINE

THE ULTIMATE SKILLS, TRICKS, AND MAKES

AVAILABLE NOW

hsmag.cc/store

FROM THE MAKERS OF **HackSpace** MAGAZINE

A pallet wood light clock

Make a DotStar-powered wireless clock using scrap materials



Dr Andrew Lewis

@monkeysailor

Dr Andrew Lewis is the owner of **Shedlandia.com**, a restorer of old tools, a fabricator for hire, a research scientist, and a founder member of the Guild of Makers.

Above
You can see from the position of the lights that this photo was taken at 7:45. This LED clock circuit could also be worked into a shoji screen, a doorway, or a window frame

Making use of scrap material is always good practice. It's especially difficult to get hold of raw materials for making at the moment, and it's also difficult to get rid of those bits of scrap that you wouldn't normally consider using. In this project, you'll see how some scrap wood and an old cast-iron ornamental panel can be used to make a frame for an unusual clock with a NodeMCU and some LED strips.

This project begins in a very Gordon Freemanesque way, with a big crow-bar and some brute force. Attacking wooden pallets to salvage wood is not the most pleasant job in the world, and the rewards can be quite variable. Sometimes the wood will split or break when you try to remove it. This is normal – you can't save every plank. The recovered wood will be used to make a frame around a metal centrepiece, so the exact dimensions will be driven by the centrepiece item that you're using.

Once you've liberated your raw materials, plane and sand the wood to remove any rough spots, dirt, and knots, and then smooth the corners. Ideally, you will need room for at least one metre of DotStar or NeoPixel LEDs on each side of the centrepiece, so try to keep this in mind when you're sorting through your scrap bin.

Build the frame around your centrepiece, using metal corner plates and glue to hold everything together firmly. You can add two hinged legs at the back of the frame if you're making a floor-standing clock, or add fixings to hang the frame if you're wall-mounting it. We would exercise extreme caution if choosing to wall-hang the frame, as it will probably be very heavy when it's complete.

IT'S PALLET TIME

With the wooden frame complete, you can mount your LEDs inside the centrepiece. The LED strips should be placed on the longest sides, so that they shine across the gap in the middle of the frame. Hot glue will hold everything in place. We used two one-metre lengths of DotStar LEDs, joined into a continuous strip by jumper wires at one end. →

ALTERNATIVES

We know what you're thinking, but if you don't happen to have a handy cast-iron table or bench panel lying around, you can still make this project using whatever bits of scrap you do have access to. A coarse material screen would work, as would some bamboo canes or willow whips, old kitchen utensils, or even a trellis with living plants.

YOU'LL NEED

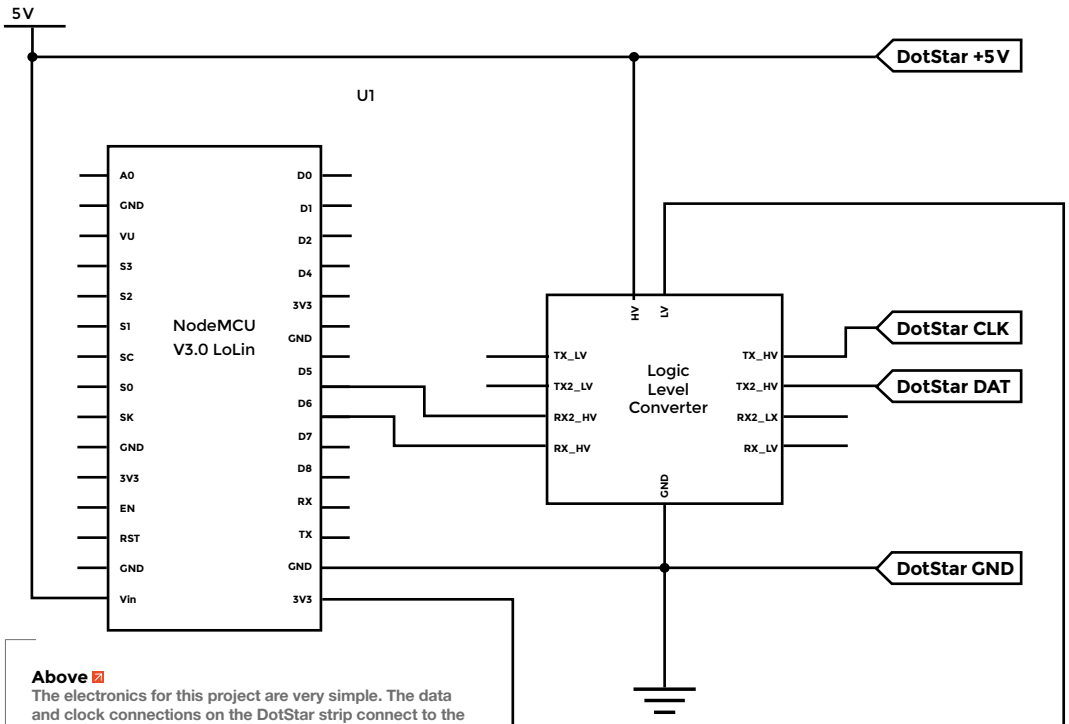
- **NodeMCU**
- **2 m of DotStar LED strip**
- **Level shifter module**
- **5 V USB adapter**

Below
You can split and join DotStar and NeoPixel strips quite easily. It's even possible to leave quite large wire sections (up to about 200 cm) between the cut sections of the strip without running into too much trouble. However, it's worth adding a power connection every 100 cm along a strip to keep the LEDs looking bright and sparkly. If the strip gets too long without an additional power input, then the LEDs will start to get dim towards the end



QUICK TIP

DotStar and NeoPixel strips have an arrow on them indicating the direction that the signal is being transmitted. Connect the strip to the NodeMCU before the first arrow, otherwise, nothing will light up.



Above
The electronics for this project are very simple. The data and clock connections on the DotStar strip connect to the NodeMCU using a level shifter module to match the strip's 5V logic to the NodeMCU's 3.3V logic. You might be able to miss out the level shifter and control the DotStar from 3.3V logic, but it's hit and miss about whether this will work



Right
Getting the numbers in the right position on the wood is easier if you modify the code to show all of the hour markings at once. You can count the LEDs manually to get the same result, but having the lights showing makes it easier to spot any mistakes before you make them

MAP TIME

Mapping the time to a strip of LEDs is easier than you might think. The Arduino language uses the `map(value, fromLow, fromHigh, toLow, toHigh)` function to translate a value from one integer range to another. With a strip of 60 LEDs, you can translate the time in hours by using the function like this: `map(hour, 1, 12, 0, 60)`. However, this will only work for a 12-hour clock, so some formatting of the time is needed beforehand. If you're using a DotStar with 120 LEDs per metre, simply change the value in the map function from 60 to 120, and the program will adjust to compensate for the change.

KEEP THINGS LIGHT

The clock display is a bit difficult to decipher without any indicator text, so you should mark the outside of the frame with numbers for hours and minutes. You could do this free-hand with a paintbrush, with transfers, or use a wood-burning pen. We opted to 3D-print some templates, cover them with thick foil to protect them, and then use them with a small butane torch to burn the numbers into the wood.

Once this is done, you can varnish or oil the wood to seal the surface and make it more durable. The basic clock is complete, so you can start to get more creative with the code. You can dowload the Arduino sketch from **hsmag.cc/issue31**. The minute side of



Left
When you're working with recovered materials, it really pays to experiment with some scrap material before you try something that you're uncertain about. While the material might not be expensive in the conventional sense, it may be difficult to replace if you make a mistake. I'd never used plastic templates for pyrography before, so I double-checked the idea would work on some scrap material before I set fire to anything important

the clock has 60 LEDs that can display more than just the time. Why not connect a 10 kΩ thermistor to the analogue input of the NodeMCU, and set an LED to display the current temperature? Another possible upgrade is setting up a simple web server on the NodeMCU to control the colour of the lights using IFTT. You could display incoming emails, the time of the next bus, high tide times, and any number of other pieces of information.



LED CHOICES

DotStar and NeoPixel strips serve a very similar function, but operate in different ways. For this project, either type of strip would have been fine to use. So what are the differences between the two types of strip? The most noticeable difference is that the DotStar needs two wires to control the LEDs, as opposed to the NeoPixel, which uses only one control wire. This is because the DotStar strip has an external clock signal, while the NeoPixel uses a fixed 800kHz data rate. The external clock means that the DotStar is more widely compatible than the NeoPixel, and can refresh the LEDs at a much faster rate than the NeoPixel. NeoPixels generally cost less than their DotStar equivalents, and are more commonly seen in projects because they're available in a wider variety of form factors than the DotStar LEDs. The LED strips also use different control libraries on the Arduino. The aptly named Adafruit_DotStar library and Adafruit_DotStarMatrix libraries control the DotStar, while the Adafruit_NeoPixel controls the NeoPixel.

REAL TIME

Using something like a NodeMCU or an Arduino as a clock has a few unique challenges, because these devices don't have a real-time clock (RTC) built-in. If you cut the power to your project, the time and date would reset. There are a couple of ways you can solve this problem – by using an external RTC module, or by getting the time remotely from a trusted source such as a radio clock. With a NodeMCU, you can use WiFi and Network Time Protocol (NTP) to get the time from an NTP server. This project uses the AceTime library to get the time from an NTP server and adjusts it for the time zone (including British Summer Time). There are other libraries available on the Arduino to deal with time, but the AceTime library is relatively abstracted and easy to use.

QUICK TIP

If you're going to modify the project to light up lots of LEDs at the same time, remember to use a suitable power supply. Allow about 60 mA per LED, and add a 1000 µF 6.3 V capacitor to the power line to smooth out current changes.

TUTORIAL

Design your own planter with Tinkercad

3D design doesn't have to be hard



Ben Everard

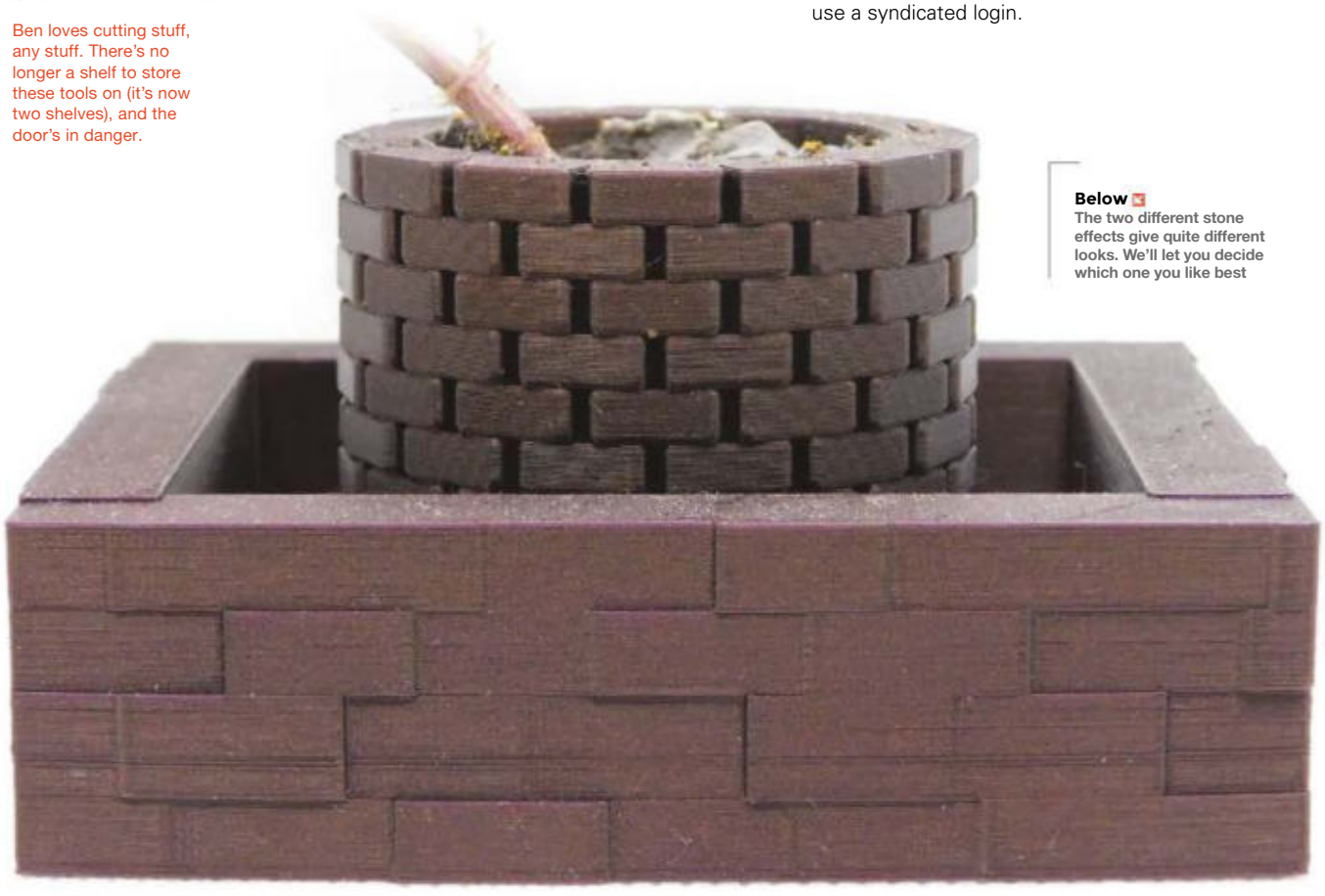
@ben_everard

Ben loves cutting stuff, any stuff. There's no longer a shelf to store these tools on (it's now two shelves), and the door's in danger.

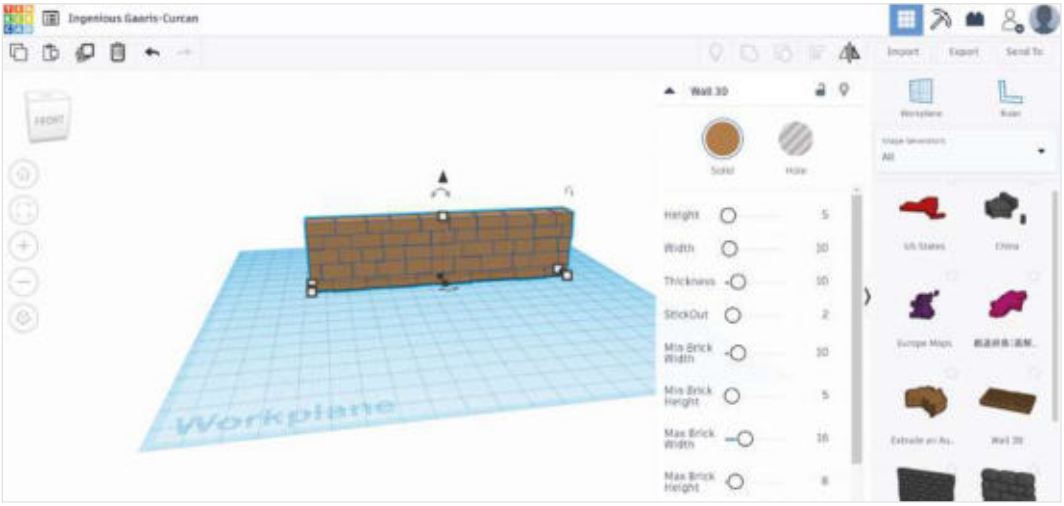
Tinkercad is a web-based 3D design tool that's easy to use, but can still generate interesting and unique patterns. In this tutorial, we'll explore a few of its features as we build up a small plant pot that's designed to look a little like a castle. This will let us explore basic shapes, generators, and the Codeblocks method for making more complex shapes of our own.

The basic design is simple. We'll have a cylindrical tower in the middle, and this will be our planter. Around that, there will be a base and a small wall which is designed to catch any water should we add too much as we water. On both of these walls, we'll have a brick effect to show off different ways of introducing texture to Tinkercad projects.

The first step is to log into Tinkercad. Head to **tinkercad.com** and either register for an account or use a syndicated login.



Below
The two different stone effects give quite different looks. We'll let you decide which one you like best



Left
You can adjust the properties of the stone wall to get the look you want

GET DRAWING

Let's start with the outer section of the planter as this is simplest.

Tinkercad works with shapes. Each shape is a 3D object, and many have parameters that you can tweak to change its size and appearance. To create the walls that will go around our castle planter, we'll use a shape generator.

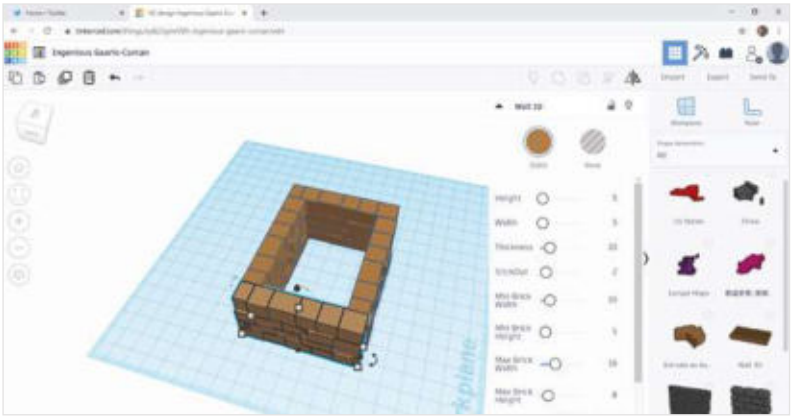
In the drop-down on the right that says Basic Shapes, change this to Shape Generators > All, and find Wall 3D. There are a few other wall options, but this is the one that we found to be most waterproof, so it's the one we'd recommend for the outer edge.

Drag and drop four of these wall 3D shapes onto the build area. Moving things around the Tinkercad build area can be a little confusing, depending on what angle you're viewing it at. Selecting the object and moving the mouse will move it in the X and Y directions, while selecting the cone above the object will let you move it in the Z direction. You can also rotate the objects by clicking and holding on the rotate arrows. If you keep the mouse inside the protractor that appears, the object will rotate in steps; if it's outside, it'll rotate freely.

> "If you keep the mouse inside the protractor that appears, the **object will rotate in steps; if it's outside, it'll rotate freely**"

You can also change the view of the build plate by clicking and dragging the cube in the top-right. Before getting into the design proper, it's worth spending a little time moving things around and getting a feel for this.

To make our planter, we obviously need the walls the right size, and we can adjust this in the settings for each object. The main ones we want to adjust are the height and width. We created two walls that were each 4 blocks high by 7 long, and two that were 4 high by 5 wide. Arrange these into a roughly square shape, with the longer sides being on the outside of the corners.

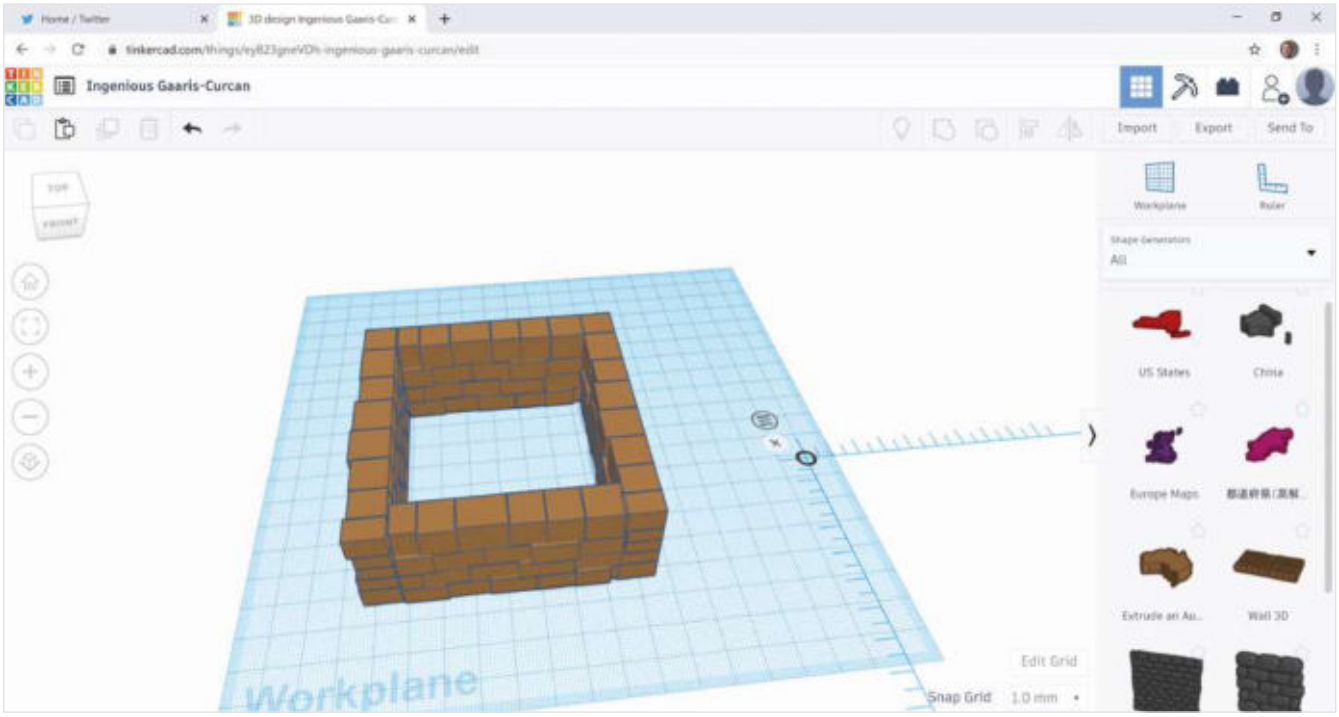


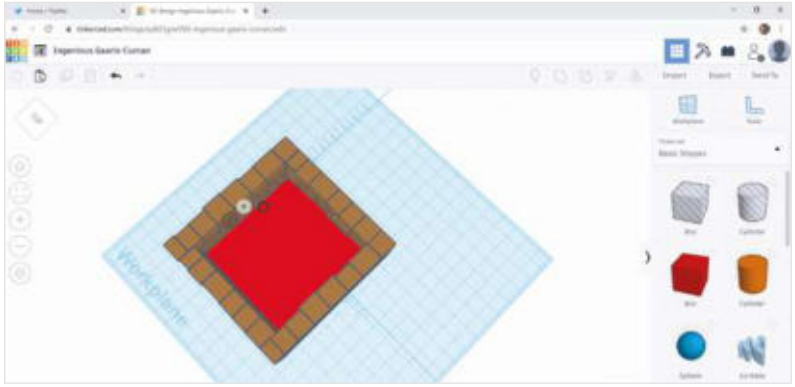
Above
Don't worry too much if your walls overlap: this doesn't affect the printing

Be careful because the walls do have an inside and an outside – though it's a bit hard to see, the texture of the bricks is only on one side.

Now we've got four walls, it's time to add a base. This can just be a box scaled to the right size, so in the drop-down on the right-hand side of the screen, select 'Basic shapes'. This should change the object list to include a box. Add one of these to your build plate, then drag and scale it so that it covers the base of the castle. It should overlap with the walls slightly – this isn't a problem as Tinkercad will output a single object that's the total of all the parts, and overlaps will simply be ignored. A slight overlap will ensure a solid part so there are no gaps. →

TUTORIAL





Top
We have two longer walls, and two shorter walls – together these make an approximate square. However, you can put yours together however you like

Above
Add a box to the bottom of your walls to form the base

BUILD WITH CODE

Now it's time to build our tower. We wanted to make a circular tower, but unfortunately, there's no way of curving the brick wall around. We have to make our shape from scratch. We could make a block for each brick and carefully place these as we go, but that's a very time-consuming thing to do, and fortunately, Tinkercad includes a small scripting language called Codeblocks that makes it easy to do things like this.

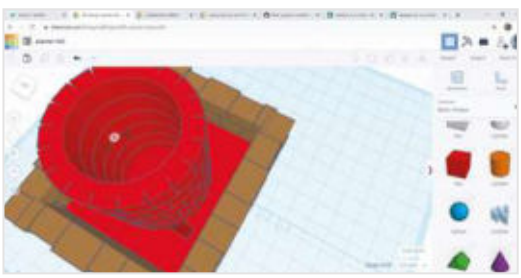
If you click on the Tinkercad logo in the top-left, you can then select Codeblocks and create a new code block, (we'll later use this to create a part that we can import into our main Tinkercad design).

You can drag-and-drop bits of code to create your script. The full code is given in **Figure 1**, and you can access the design online at **hsmag.cc/iJEIwD**. We won't go all the way through it, but briefly:

- The inner loop creates a circle of bricks. It calculates their position using sin and cos functions. As well as a position, each brick has edge = 1; this gives it rounded corners, which makes it easier to print and looks better. Each brick is also rotated so that it faces outwards.
- The outer loop creates each layer of bricks. The `rotate-step` variable is used so that each layer is half a brick out of step with the layer below, and this creates the brick pattern.

If you want to create towers with different widths, you can play with the `radius` and `num_bricks` variables. This code isn't intelligent enough to work out how many bricks of a given size are needed for a given radius – we'll leave that as an exercise for the reader. To run the code and generate the design, click on the

Below
The negative cylinder lets water drain out of the base of our planter if we water it too much



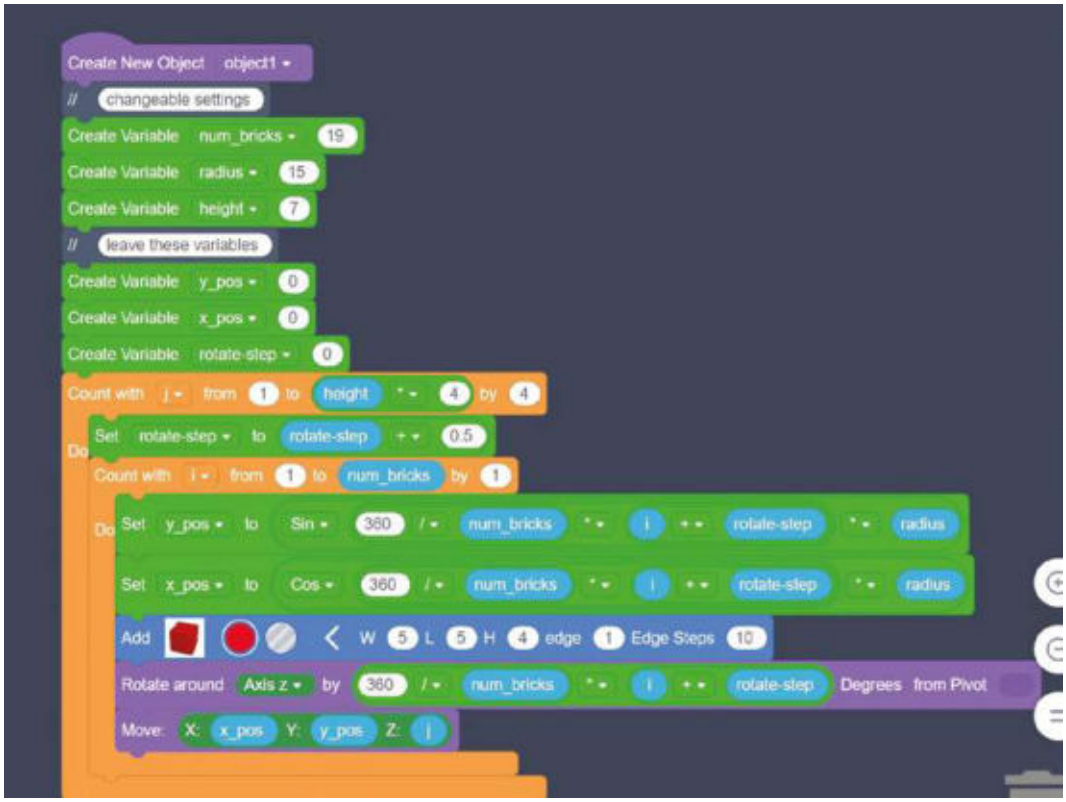


Figure 1
A pair of loops place all our bricks in the right position to make our tower

triangle in the top-right. It can be a little slow, so you might want to ramp up the speed slider.

Once you're happy with the design, click on Export > Part, and this object will now be available in Tinkercad. On the Shapes drop-down, scroll down to You > Parts collection, and you can add your tower like any other part. We scaled ours to almost fill the inner square, and up high enough to tower over the walls (if you'll excuse the pun).

NEGATIVE SPACE

We've almost finished our planter now, but there's one more thing needed. We wanted a way of water draining out of the main planter into the surrounding area. The shapes we add can either be solid objects or holes. If you add a hole, this will, unsurprisingly, add a hole to an object. Add a cylinder to your project and, in the properties, set it to be a hole. You can then rotate it so that it's horizontal, size it appropriately (we had a diameter of 5), and position it so that it goes through the base of the tower into the moat area.

That's all there is to it. If you want to print your planter, you will need to export it. Hit the Export button in the top-right. Make sure you check the box to export everything (rather than just the currently selected part), and select STL (or whatever format you like). You can then slice and print this as usual.

OTHER DESIGN OPTIONS

Tinkercad is easy to use and does let you create useful models, but it is limited compared to some more fuller-featured CAD tools. When working with 3D modelling software, bear in mind that the tasks are fundamentally complex – and that means that user interfaces and workflows can seem a bit convoluted, particularly at first. Spending a little time getting to know the different options, before selecting the right one for you to invest time in learning, will pay off later.

A few options you might want to consider are:

- OpenSCAD is similar to Codeblocks in that you use code to create your shapes rather than interacting with the mouse, but it's far more powerful. If you enjoy the programming approach to design, then this is one you should consider. We'll be looking at it more closely in a future issue.
- Fusion360 is probably the most popular 'fully featured' CAD tool around. However, it is a commercial tool and is only free for one year of personal use. After this, it is quite expensive.
- FreeCAD is an open-source mechanical design tool. It can take a bit of time to learn, but it has a lot of features and can be used to create complex models.

TUTORIAL

Make noise with CMOS

Build a square wave oscillator for beautiful DIY music



Jo Hinchliffe

@concreted0g

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY space. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open source. He also has a shed full of lathes and milling machines and CNC kit!

A **fascinating area of lo-fi electronic music is the CMOS noise-making community who often build complex instruments that many call Lunettas.** The name arises from one of the earliest people to do this, Stanley Lunetta, a percussionist, avant-garde musician, and performer from New York who built all kinds of noise-making contraptions and synthesisers in his lifetime. CMOS stands for Complementary Metal Oxide Semiconductor and often refers to a family of chips that perform common logic functions such as latches, flip-flops or (as in this case) Schmitt triggers. More specifically, this type of chip is known as 4000 series CMOS as the chip name is usually a number in the 4000's (though can be in the 40000's). While these chips weren't designed for making audio, they interact with signals fast enough to manipulate audio in stange and unusual ways.

There are no strict rules when building Lunettas – it has some connections to circuit bending, and some of the founding principles there are worthy of note. Using low-power battery supplies until you know enough to confidently look at other power supplies, and not connecting your first creation to a high-end stereo are both good rules to follow here! A battery-powered small speaker/amp can be a good project in itself, and there are numerous kits out there – or indeed there are these cheap and cheerful micro guitar amps.

In **Figure 1**, we can see 'Blarp' which is a DIY-designed modular 'Lunetta' synthesiser (modular

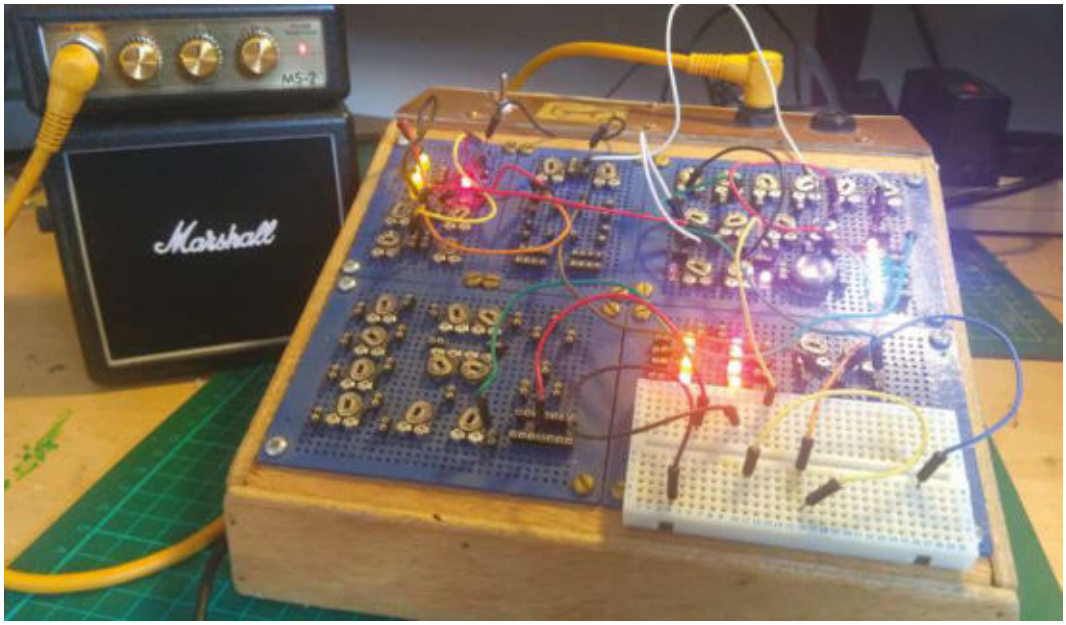


Figure 1
A miniaturised modular CMOS Lunetta noise-maker 'Blarp'

meaning it has a heap of different circuits that can be connected in different ways). Blarp was made on a tight budget and used round pin headers instead of the 4 mm banana jacks (which are more common in synths), and trimming potentiometers (trimpots) rather than full-size potentiometers. This meant it had to be 'played' using a plethora of tiny flat-head screwdrivers, but also meant the design could be shrunk to its small scale. If the hundreds of connections had used full-size connectors, it would have been the size of a small chest of drawers! While Blarp included many circuits, sequencers, dividers, weird modulators, gated oscillators, and envelope generators, we are going to start with a basic building block – some voltage-controlled square wave oscillators.

LET'S GET BUILDING!

We are going to use a classic CMOS 4000 IC – the CD40106 hex Schmitt trigger inverter. It's incredibly useful and can be used to create six voltage-controlled oscillators (VCOs) which primarily can be used and blended together to make all kinds of bleepy, droney noises. Individual oscillators can also be used as clock signals to control the tempo of other CMOS/Lunetta noise modules. Experimentation is to be encouraged

> Experimentation is to be encouraged – **don't worry if you don't have the exact values of components**

– don't worry if you don't have the exact values of components – trying different values, again, sticking to our 'everything is battery-powered' rule is definitely the order of the day.

Grab a breadboard and carefully insert your CD40106 IC across the gap to break out each pin. Looking from above, with the notched end of the CD40106 uppermost, the pin numbers start with 1 on the upper left and work down and around to pin 14 on the upper right as seen in **Figure 2.** Pin 14 is the positive power connection, and pin 7 is negative/ground. Pins 1 and 2 are the first inverter pin pair that we are going to set up as our first oscillator.

Following the schematic shown in **Figure 3**, connect Pin 1 to the left-hand pin of our trimpot. We also want to place our 0.1 µF capacitor between pin 1 and ground. Next, wire the wiper of the trimpot (centre pin) to pin 2. That's our first oscillator patched! →

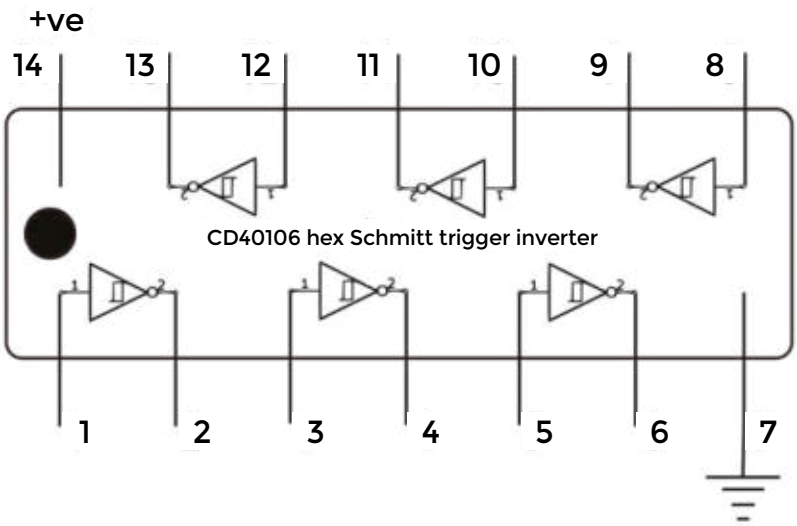


Figure 2 The CD40106 with its six inverters. The inputs to each inverter are all labelled 1, and the outputs 2

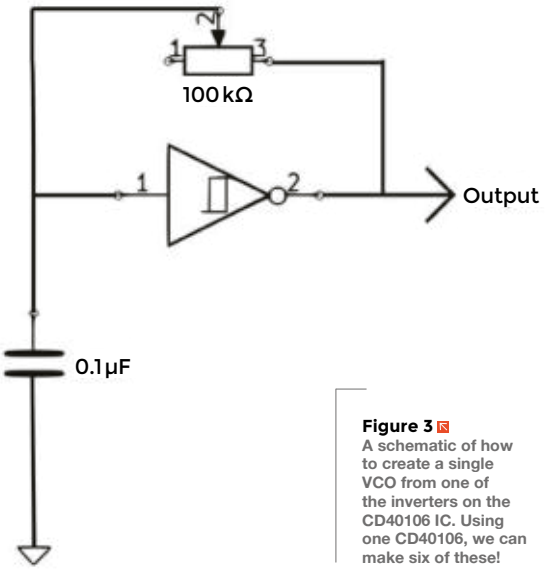


Figure 3 A schematic of how to create a single VCO from one of the inverters on the CD40106 IC. Using one CD40106, we can make six of these!

MIXED SIGNALS

Mixing through resistors is a type of unamplified or passive mixing – and it's pretty simple to make a variable volume passive mixer using a single resistor and a variable resistor or, you guessed it, a trimpot! **Figure 7** shows a simple schematic for a passive variable mixer – these can be built on a prototyping board. Experiment with different values for the variable resistors and the summing resistors, but as a rule, they should match. So if you choose a 10 kΩ trimpot, you should pair it with a 10 kΩ resistor. It's important with these mixers – similar to the jack socket direct output – that the ground connection from other modules is carried through to the output jack, so you must make sure that you include a way to patch the mixer to ground. In **Figure 1** on the 'Blarp' modular build, the upper right-hand corner features eight passive mixer channels.

YOU'LL NEED

- **CD40106 hex Schmitt trigger inverter IC**
- **Breadboard and wires**
- **A battery-powered amp and speaker**
- **9 V battery**
- **10 kΩ resistors**
- **100 kΩ variable resistors** (trimpots are fine)
- **0.1 µF capacitor**
- **1N4148 signal diodes**
- **Some other value capacitors and variable resistors for experiments!**

QUICK TIP

Start with the volume low on the amp – be aware that the volume of the oscillators can suddenly increase!

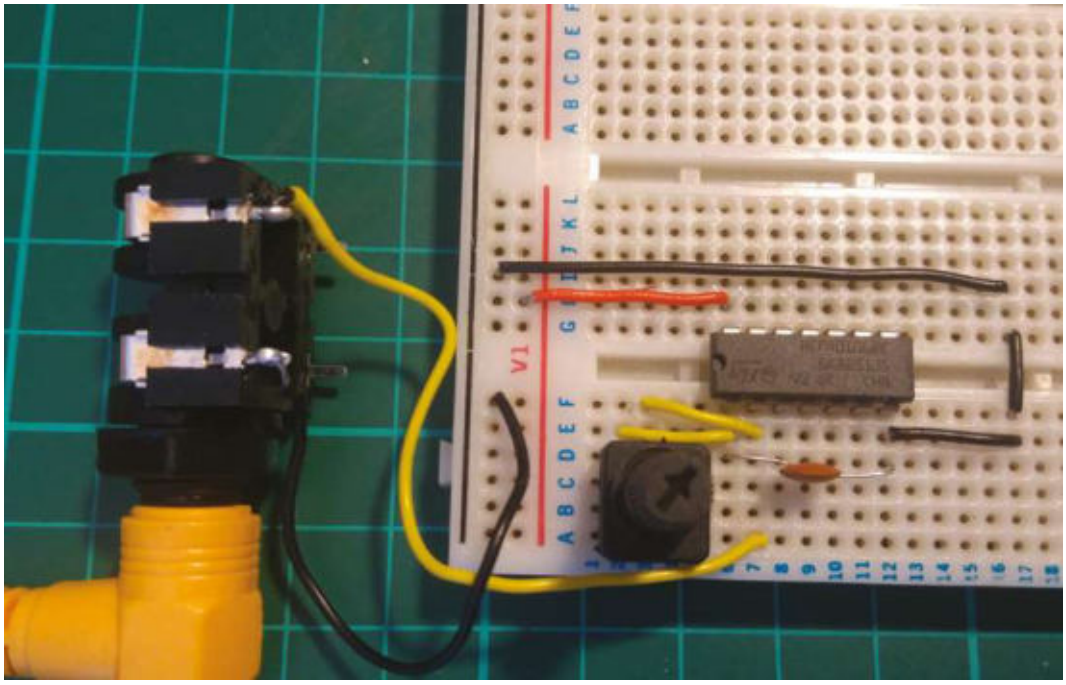


Figure 4
First VCO patched up on a breadboard and a jack socket to take the signal to our battery-powered amplifier

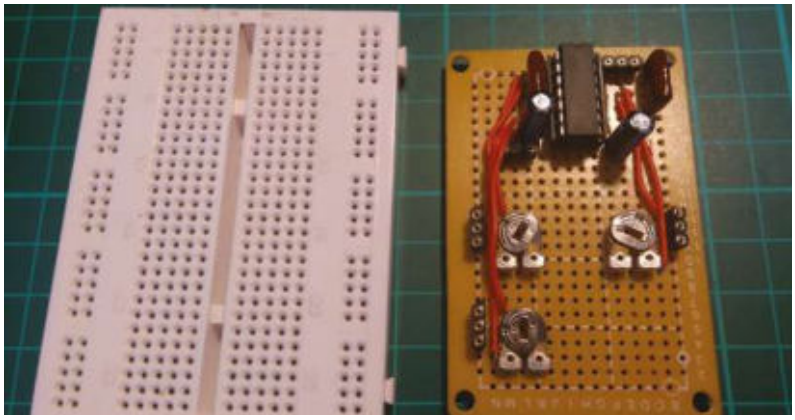


Figure 5
We've permanently soldered up three VCOs on a small prototyping board, with connections broken out to allow us to use it with other modules

Grab a jack plug socket and solder some solid core wire to the ground connection and the signal connection. A good way to work this out is to use a continuity tester on a multimeter. Plug a short jack lead into the socket and test between the unplugged jack plug tip and ground area and the contacts on the socket end to see which connect. Once you have this, connect the signal wire you just soldered to pin 2 and connect the ground wire to the ground on the board (see **Figure 4** overleaf). Connect the jack to the battery-powered amp, and then connect the power to the circuit by connecting the battery to the breadboard.

GLORIOUS NOISE!

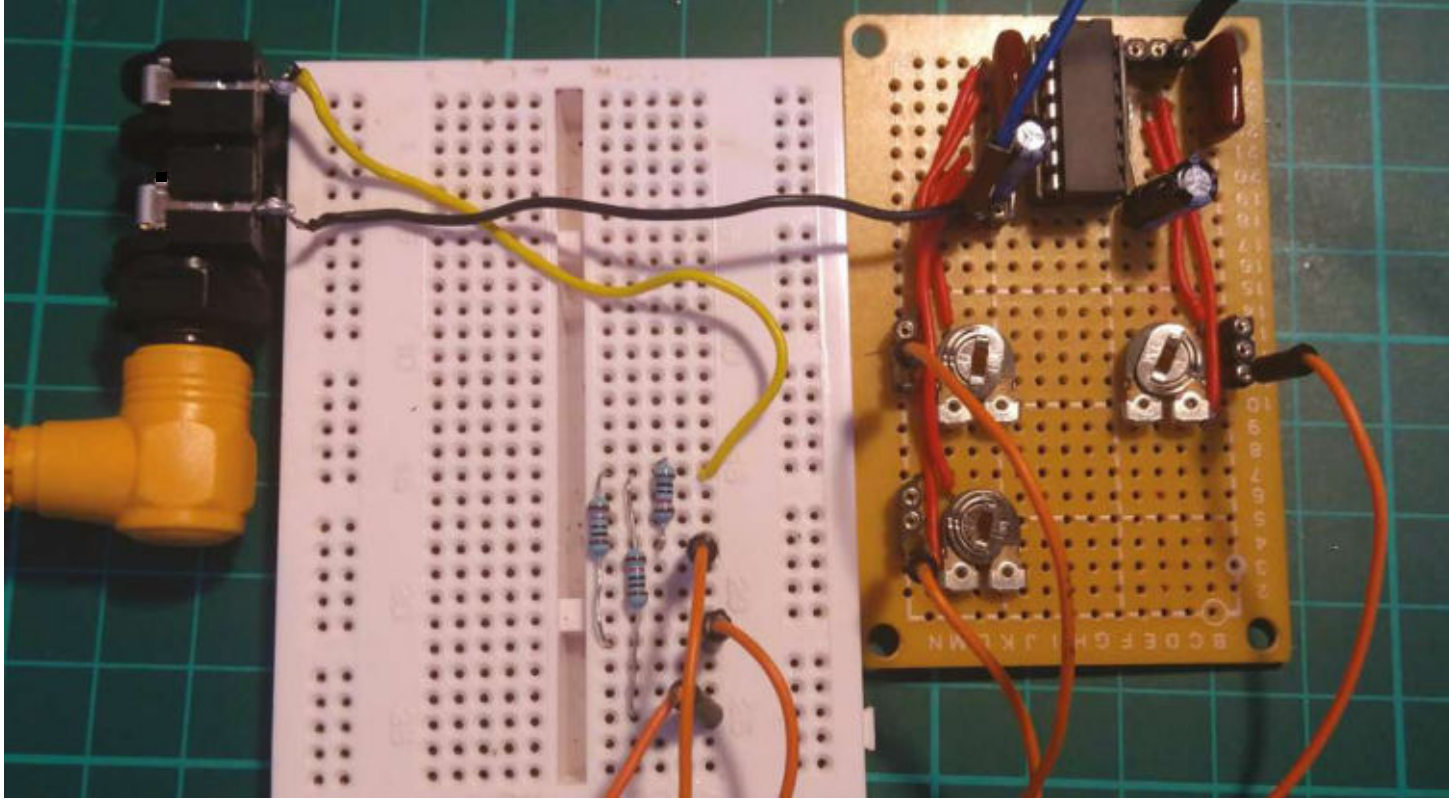
You should now be delighted to hear the warbling of a square wave that you can change in frequency using the trimpot. Of course, the next task is to wire up more/the rest of the oscillators. You can continue on the breadboard, or you might prefer to jump to soldering the circuit onto perfboard or Veroboard or, of course, some creative point-to-point wiring or dead-bug style – whatever takes your fancy.

In **Figure 5**, we have soldered up three oscillators using three of the available six inverters of a CD40106 IC onto some prototyping board. We have used the extremely cheap metal-pressed trim potentiometers, and some scrap round pin headers as connection points.

FUTURE CONNECTIONS

We built this in a way that allows it to be used in a modular fashion, so we have left multiple power and ground pin headers exposed, as well as a three-pin header as the output of each oscillator. This means that we can daisy-chain power to and from the board to the next/previous module, and we can also send the oscillator outputs to different places. For these oscillators, after experimentation, we decided to use 1 MΩ trimpots for all of them, and we set two of the oscillators up with 1 µF capacitor and the other with a 0.1 µF capacitor. Note that we have also added a 0.1 µF decoupling capacitor across the power and ground rail which is good practice to do near the power pins of each CMOS module you build. It helps even out the power supply and provides some reserve current in the capacitor near each IC, and generally makes the CMOS chip behave as you'd expect it to.

Now we have three oscillators; we need a way to mix them together. The first and most common way



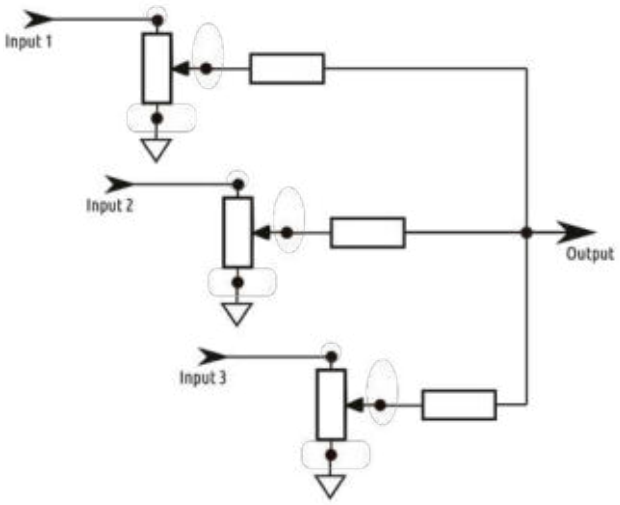
is to mix each one through a resistor (making sure each resistor matches). In **Figure 6**, we can see that we have taken a wire from each output to a breadboard and added three 10kΩ resistors – then connected the mixed output to our signal wire on the jack socket, and the ground jack socket wire to the ground pin headers on the oscillators board. If you connect power and turn on your amp, you should now hear the cacophony of three oscillators, whose frequencies you'll be able to mess aroiund with, mixed together!

Another experiment is that instead of mixing the signals together with resistors, we can use small signal diodes to combine the oscillator outputs. This is not a correct way to do this in audio terms, but it does create a type of 'modulation' – again, this isn't the fancy, proper modulation a true modulator circuit/device would perform. Simply sum the three oscillator outputs using small 1N4148 diodes, and then connect this combined output to the jack socket signal wire. When you power on the circuit and the amp, you should find that the sound is potentially more 'rhythmic', and often it seems like one oscillator is turning another oscillator off and on.

A final thing to do is to record your fine, noisy opus for all to enjoy. We would urge that, until you are sure of what you are doing, it's probably best not to hook this up to an expensive sound card or mixing desk. A good starting point can be to simply record using a microphone/dictation machine set up in front of your battery amp. Once you have some digital files of your noise-maker, it can be great fun to try and add effects, reverbs, delays, filters, and others using audio software such as the free, open-source Audacity – available here: **audacityteam.org**. Finally, we have merely scratched the surface of Lunettas, and we heartily recommend you look around at what others have done in the community – a good starting point is the Lunettas forum on the electro-music forum here: **hsmag.cc/e9jvDk**.

Figure 6
Mixing the signal outputs together using some matching resistors

Figure 7
A simple mixer for combining multiple audio outputs



PULL **DOWN**

Another good rule to follow for modular CMOS noise-makers is that, if you are connecting the output of one logic system (or an output of a gate/inverter, for example), it can be directly connected to an input of another logic gate. However, if you are leaving an input unconnected or 'floating' on a CMOS chip, it's a good idea to connect a resistor (10kΩ seems a common choice on Lunetta-building forums) between the input pin and ground (pull-down), or between the input and positive (pull-up). These pull-down or pull-up resistors, again, similar to decoupling capacitors, can cure a lot of stability problems. If you are building a collection of modules that you want to be able to 'patch' together in different configurations, it's a good idea to put these resistors on all inputs so that whether they are used or not in a future patch, they will be in a stable state.

DON'T MISS THE **BRAND NEW** ISSUE!

SUBSCRIBE FROM JUST £5

- **FREE!** 3 issues for the price of one
- **FREE!** Delivery to your door
- **NO OBLIGATION!** Leave any time

FREE PI ZERO W STARTER KIT*

With your 12-month subscription to the print magazine

magpi.cc/12months

* While stocks last

Buy online: store.rpipress.cc

HackSpace
TECHNOLOGY IN YOUR HANDS

FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG 106

CAN I HACK A C64



A retro computer lovingly poked and prodded



PG 100



BEST OF BREED

The finest maker kits to help you learn new skills

REVIEWS

108 **Enviro+ FeatherWing**
Link your Feather to the weather

110 **High Quality Camera**
A new flexible imaging module



112 **Ultimate maker case**
3D-print a parametric box



113 **Book review**
Micro:bit for Mad Scientists

ONLY THE BEST

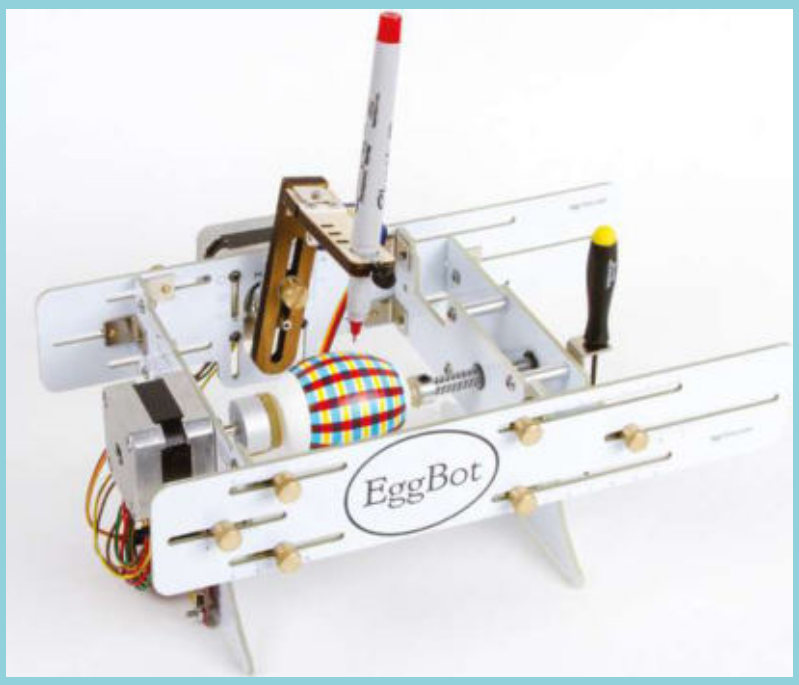
Kits that will inspire you

Sometimes you just need someone else to figure out the project!

By **Marc de Vinck** @devinck

Sometimes you just need someone else to figure out your next project, and that's where this roundup comes into play. Don't get us wrong – we love to think up new projects and build them from scratch. But then there are times where it's just fun to assemble something, and not worry about if it will work! In this Best of Breed, we'll be looking at a few of our favourite kits for anyone looking to build a project, but not looking to figure out all the minutiae and complexity associated with creating electronic projects from scratch. No, not that Scratch!

One criterion we have for defining a great kit is the ability for the project to be long-lasting. Not that the build is long, but rather the user experience after the build. A usefulness, whether it be playful or productive, for the end product and its user. We've all seen kits that are fun to assemble, but once completed, end up in a drawer somewhere or, worse, a landfill. All of the kits that we're looking at today can be enjoyed for a long time, even after the joy of building it is over.



Adafruit IoT Pi Printer Project Pack vs Solarbotics Sumovore Mini-Sumo

ADAFRUIT $115 | adafruit.com **SOLARBOTICS** $98.50 | solarbotics.com

This is one of our favourite kits from Adafruit. It's fun, easy to build, and really useful. It's also a great introduction to building an Internet of Things device. Not only is it connected, but it generates physical media via a Raspberry Pi and the kit's built-in mini thermal printer. You supply the Raspberry Pi – the kit contains everything else needed.

Wondering what you can connect to and print? Try a daily weather forecast report, Sudoku puzzles, or an inspirational image from your favourite Twitter feed. It's all possible thanks to the open-source code and detailed tutorial found on the Adafruit project page. You'll need some basic soldering skills, but even a beginner should be fine with its assembly. So get building, get online, and get printing!



Left Turn the digital world into physical bits of paper

Below Challenge your friends to robotic battle



Now, who doesn't like a good old-fashioned robot battle? That's right, nobody! And that's why we're fans of the Solarbotics Sumovore Mini-Sumo robot. Unlike many other robots out there that add complexity with microcontrollers, the Sumovore uses a 74HCT240 octal inverting buffer chip as its brains. It's similar to BEAM-style robots, but this is a bit more advanced. Through some simple switches, you can change the behaviour of the Sumovore. And for those of you who want to add a microcontroller, it's ready for that too!

Once you complete the kit, there are lots of optional things you can add, including breakout boards for adding a PICAXE, BASIC Stamp, or other microcontrollers. And you can even increase the speed and torque by swapping out the standard motor for a new one. You'll need to take a look at the online instructions, and incredibly detailed descriptions, on how this little sumo bot works to understand all its features. It's a fascinating read.

VERDICT

IoT Pi Printer Project Pack

A fun home for your Raspberry Pi.

9/10

Sumovore Mini-Sumo

Think about picking up two for much more fun!

9/10

SparkFun Simon Says

SPARKFUN $27 | sparkfun.com



Right
Just press the buuttons in the right order

We all know this classic game just by looking at the four coloured buttons. The SparkFun Simon Says kit is a simple and fun soldering kit that goes along with the theme of kits that are fun to use, even after you build them. This is the newest version of their through-hole kit. They made changes to make it even easier for beginners to assemble.

The game is based on the ATmega328, which comes preprogrammed with Simon firmware. There is also a buzzer, four LEDs, button pad, battery clips, and all the necessary hardware to assemble the kit. No programming is required, but we suspect once you build this kit, you'll want to dive into the code and program the ATmega yourself, which if you pick up a few other accessories, such as an FTDI breakout board, you can access the microcontroller, too.

VERDICT

SparkFun Simon Says

Easy to solder and fun to play.

8/10

Solarbotics SolarSpeeder v2.0

SOLARBOTICS $27.50 | solarbotics.com

The Solarbotics SolarSpeeder kit is a simple solar-powered drag racing machine. You only need basic soldering skills to complete this kit, and it only takes an hour or so, depending on your skill level. Once completed, just leave it in the sun to charge up – and get ready to race. Speaking of racing, be sure to consider picking up two of these kits, as racing and fine-tuning is a lot of fun when there is some competition.

The SolarSpeeder features a unique flexible PCB for its chassis. This isn't something you see very often in the kit-making world – we like how they featured it in this particular kit. Solarbotics includes a great online tutorial and a satisfaction guarantee. Head on over to their site to learn more about this unusual little racer.



Above
Eco-friendly electonic racing

VERDICT

Solarbotics SolarSpeeder v2.0

An unusual PCB makes for a fun build.

9/10

Pirate Radio – Pi Zero W Project Kit

PIMORONI ◈ **$45** | **pimoroni.com**



Right ◈
Go to radiocaroline.co.uk for the full pirate experience

The Pirate Radio kit from Pimoroni is the perfect example of what we love in a kit. Fun? Yes! Includes everything you need? Check! Is useful after assembly? A big yes! This kit comes with everything you need to make your own radio. And not just any radio, but a Raspberry Pi-powered radio!

The kit includes a Raspberry Pi Zero W, the pHAT BEAT DAC and stereo amp, a 5W speaker, and a custom retro-styled acrylic enclosure. So many kits are missing enclosures, but not this one. It also includes other things, such as the USB cable for power. The aesthetics alone won us over, but after assembling this kit and playing around with the open-source software, I was amazed. You can play music from the internet, stream music from your Spotify account, or even make it an AirPlay speaker. The kit goes together in about 30 minutes, but the usefulness will last a lifetime.

VERDICT

Pirate Radio – Pi Zero W Project Kit

Another fun kit from Pimoroni.

10/10

Mood Light – Pi Zero W Project Kit

MOOD LIGHT ◈ **$35** | **pimoroni.com**



Left ◈
Set the ambiance of your desk

Now, I know what you are thinking. A mood light is useful, but how useful? Well, we're here to tell you that this mood light is *really* useful. Why? Because it's based on Raspberry Pi Zero W and the Unicorn pHAT. Yeah sure, you can make it change colour, or cycle through colours. But because you have Raspberry Pi at your disposal, you can also connect it to the internet and create a Twitter feed notification, email alert, subscriber system, or just about any simple alert system.

And just like the Pirate Radio kit, the Mood Light kit from Pimoroni includes everything you need to get started, including the Raspberry Pi Zero W, a Unicorn pHAT with 32 programmable RGB NeoPixel LEDs, male and female header pins, the light stand, micro USB cable, and more. Some soldering is required, but even someone with basic skills should be fine. Head over to the product page to check out the Python libraries, and detailed instruction for building your WiFi-capable mood light.

VERDICT

Mood Light – Pi Zero W Project Kit

A surprisingly useful mood light kit.

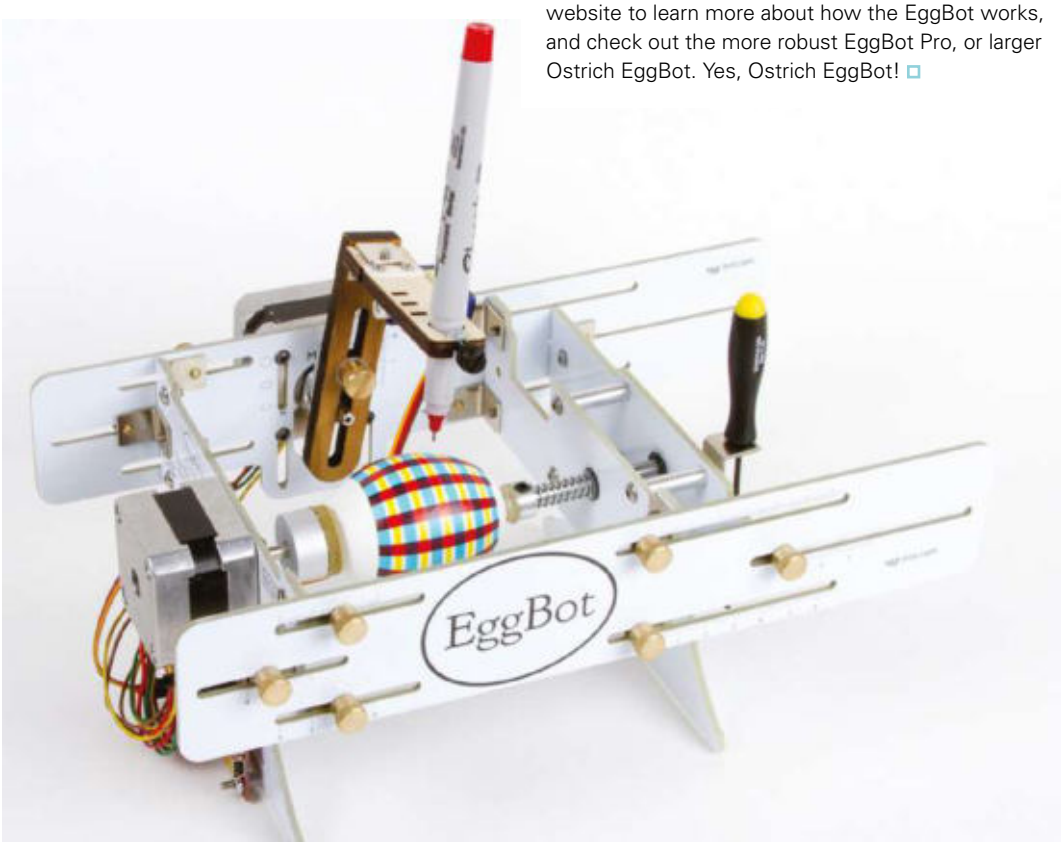
9/10

The Original EggBot: Deluxe Edition

EVIL MAD SCIENTIST ◈ **$220** | **shop.evilmadscientist.com**

Believe it or not, this author has owned several EggBots over the years. He's brought them to events, used them in class, and also used them to make some fun projects at home. The Evil Mad Scientist team based their EggBot on the original design by Bruce Shapiro, dating back to 1990. The EggBot is very adjustable, and is designed to draw many different surfaces and spherical shapes. Think ping-pong balls, light bulbs, decorations, and even wine glasses!

What we really like about this kit is the software and learning process. You start by designing something on a computer – then you use software and custom-designed plug-ins to control the stepper motors and make the bot do your bidding. It's a great introduction to the amazing world of computer numerical control (CNC) programming. It's also fun to think about modifications, like adding a diamond point for etching glass, or what about using conductive inks? Yes, you can draw on an egg, but you can also do a lot more with the EggBot! Head over to the website to learn more about how the EggBot works, and check out the more robust EggBot Pro, or larger Ostrich EggBot. Yes, Ostrich EggBot! □



Left ◈ Breakfasts need never be boring again

VERDICT

The Original EggBot: Deluxe Edition

A very well-made kit.

9/10

GET STARTED WITH ARDUINO®

Robots, musical instruments, smart displays and more

£10 with FREE worldwide shipping



Inside:

- Build a four-legged walking robot
- Create a Tetris-inspired clock
- Grow veg with hydroponics
- And much more!

AVAILABLE NOW hsmag.cc/store

plus all good newsagents and:

WHSmith BARNES&NOBLE

Available on the App Store

GET IT ON Google Play

FROM THE MAKERS OF HackSpace MAGAZINE

Can I Hack It?

A Commodore 64?

Can we reuse an old Commodore 64?



Les Pounder

@biglesp

Les Pounder loves taking things to pieces and seeing how they work. He teaches others how to be makers and tinkerers at events across the UK. He blogs at **bigl.es**.

In August 1982, a new home computer was launched from Commodore. The Commodore 64, known as the C64 by fans, brought more power to the user. The C64 was in production for twelve years and, in that time, it grew quite a fan base. This was largely due to a great selection of games and the homebrew computing groups who used the C64 to power electronics projects, write demos, music, games, and surf a large number of bulletin boards, where communities swapped messages, games, and info.

But is the C64 still any use in 2020? For that, we need to take a deep look inside the case and into the electronics which powered a generation of bedroom coders.

GENERAL CONSTRUCTION

A sturdy plastic case surrounds the C64 and provides rigidity and protection for the electronics within. The plastic can be worked with hand tools and power tools, but do take care as the plastic in our specimen is old and fragile in places. The case is held in place by three screws and a plastic hinge – again take care when opening up the case, as the hinge will be brittle.

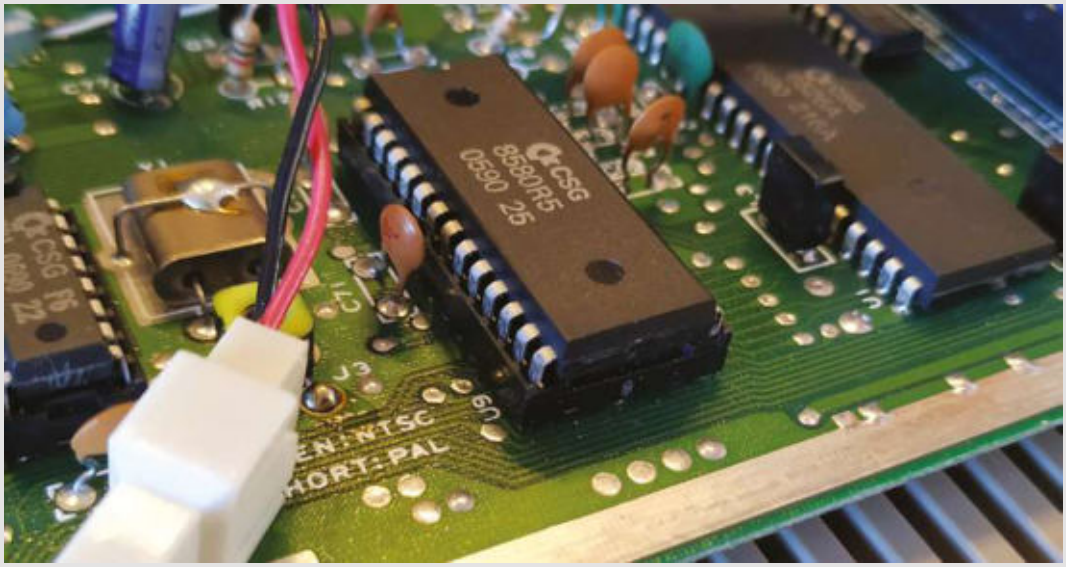
ELECTRONICS

The C64 is an interesting look back into 1980s electronics. Through-hole components are used across the board, which means we can easily repair any issues with a soldering iron and a YouTube video. The first thing to do with a 'new' C64 is to check the power supply as it's prone to sending more than 5V down the 5V DC connection, and this will kill the computer. So pick up an OpenC64Saver, or a modern PSU. The C64 mainboard came in many different versions. From the first iteration to the final one, there were changes made to reduce the cost and to swap out end-of-life components. On the main board, the C64 has a number of chips: there's 64KB in RAM chips, a CPU clocked at 1.023MHz, a VIC-II which provides graphics, but the respected and prized chip was the MOS Technology 6581/8580 SID (Sound Interface Device) chip that provided the means to make iconic music with the C64. Even in 2020, SID chips are prized for their use in music production.

We may like to think that Raspberry Pi started the trend of exposed GPIO pins in a computer, but the



Left
The Commodore 64 enabled 1980s bedroom coders to fully realise their ambitions. Oh, and the case was once a nice 'off white' colour, honest!



Left
The sound interface device (SID) chip provides the excellent 8-bit music that we love. This is a later model chip – earlier models are highly praised

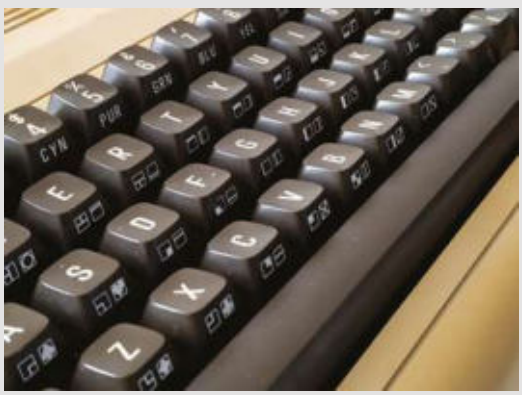
Bottom
The keyboard of earlier C64 had symbols printed on the bottom side. These symbols can be used in code to add simple graphics to text programs

> The rewards in learning this skill are great, but there is a **heavy investment of time and knowledge required**

C64, and many other computers before it, had a GPIO which a user could access. The user port on the rear of the computer provides eight GPIO pins which we can use to connect up to LEDs, motor controllers, and much more.

HACKABILITY

So how can we use the C64 in a hack? The answer is the user port! Yes, via this interface we can issue commands to control the status of the eight GPIO pins. All we need is a breakout board, available via Tindie or eBay, and a quick reference to the GPIO layout. Using commands in BASIC (Beginners' All-Purpose Symbolic Instruction Code), we can directly alter certain addresses in the C64 memory. This is called a 'poke', and to read an address we use a 'peek'. For example, to turn on pin PB0 we type POKE56577,1, and if there is an LED attached to that pin, it will light up. Going further, we can also connect the C64 to bulletin boards using a Wemos D1 Mini as a bridge between the C64 and WiFi. And, just recently, we have seen a C64 control an Arduino using a serial interface.



CONCLUSION

Right now, the Commodore 64 is enjoying a resurgence in popularity, thanks in part to an active retro computing scene, and groups of eager coders/hackers/makers wanting to push the technology as far as it can go. The rewards in learning this skill are great, but there is a heavy investment of time and knowledge required. But keep at it, and you will learn the greatness of 1980s computing.

Happy Hacking!

YOU'LL NEED
Commodore 64

COST
£ Various

WHERE
Available via eBay / thrift stores

PARTY LIKE IT'S 1989

Computing in the 1980s was an immediate and rewarding experience, but at times it was also frustrating. Needed a little help with your code? Well, there was no internet, so you had to speak with your friends, teachers, or local computer store. My journey with computers started in the 1980s with a Commodore 16; using that computer, I learnt how to type, how to play games, and how to code. Well, sort of! The first few lines of code that I ever wrote were in BASIC, and it was the classic looping 'Hello World' text. I bought magazines and typed in the code listings, to various levels of success. But it was with the C64 that I achieved more. I learnt more BASIC coding skills, but never really applied them to anything 'useful'. I had no way to store my creations, as I didn't know how to save my code to cassette tape, and floppy drives were far too expensive.

It was only when I started my new adventures with 8-bit computers via the excellent RC2014 Micro Z80 DIY board, and further with my eBay C64 and a Tindie user port breakout board, that I realised that I could create code to control electronics, save the code for future use to an SD card interface (SD2IEC) that emulates a floppy drive, and create useful projects. So, here starts my new adventure – learning to code (again) in BASIC and using it to control modern-day electronics.

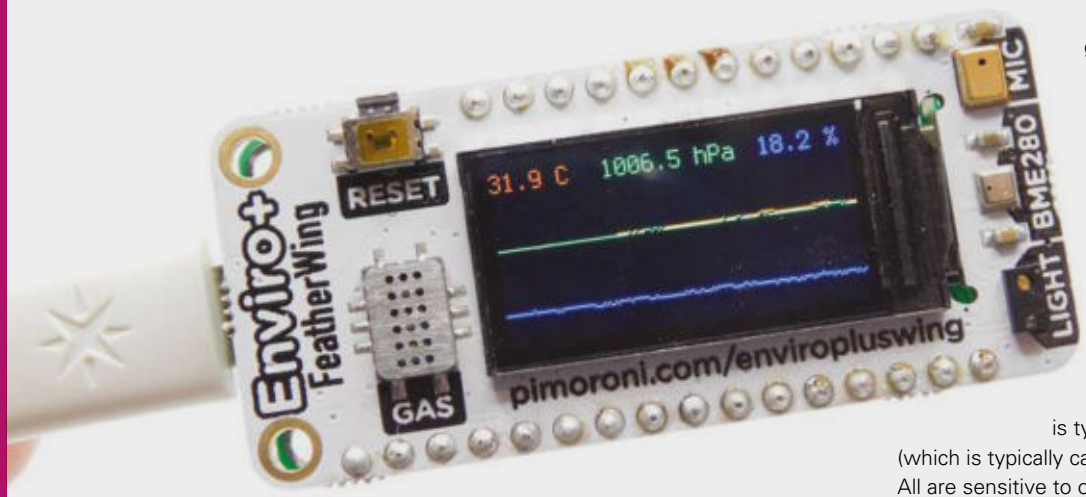
REVIEW

Enviro+ FeatherWing

Read and display data about our surroundings

PIMORONI £45 | shop.pimoroni.com

By **Ben Everard** @ben_everard



Above
White solder mask shows if you dally with your iron on!

Pimoroni have been making the Enviro + Air Quality for Raspberry Pi for a while – and now they've brought the same set of sensors and display to the Feather form factor. This gives you the ability to read temperature, pressure, and humidity via a BME280; light and proximity via an LTR-559; some gases (see below) via a MICS6814, and sound via a MEMS microphone. You can also connect a PMS5003 particulate matter sensor to detect PM2.5 and PM10 pollution (available separately). These are all well-supported sensors, and there's an accompanying CircuitPython library to get everything working together.

The quality of environmental sensors varies hugely between different models and manufacturers. There are lots of really cheap sensors available which work OK, but can't really be relied upon for any real accuracy, and there are also hugely expensive scientific instruments that are sometimes needed. These sensors sit somewhere in the middle. They're good for what they're intended – monitoring the environment. They're accurate enough that you can be confident of the readings, but not so costly as to make the board ruinously expensive.

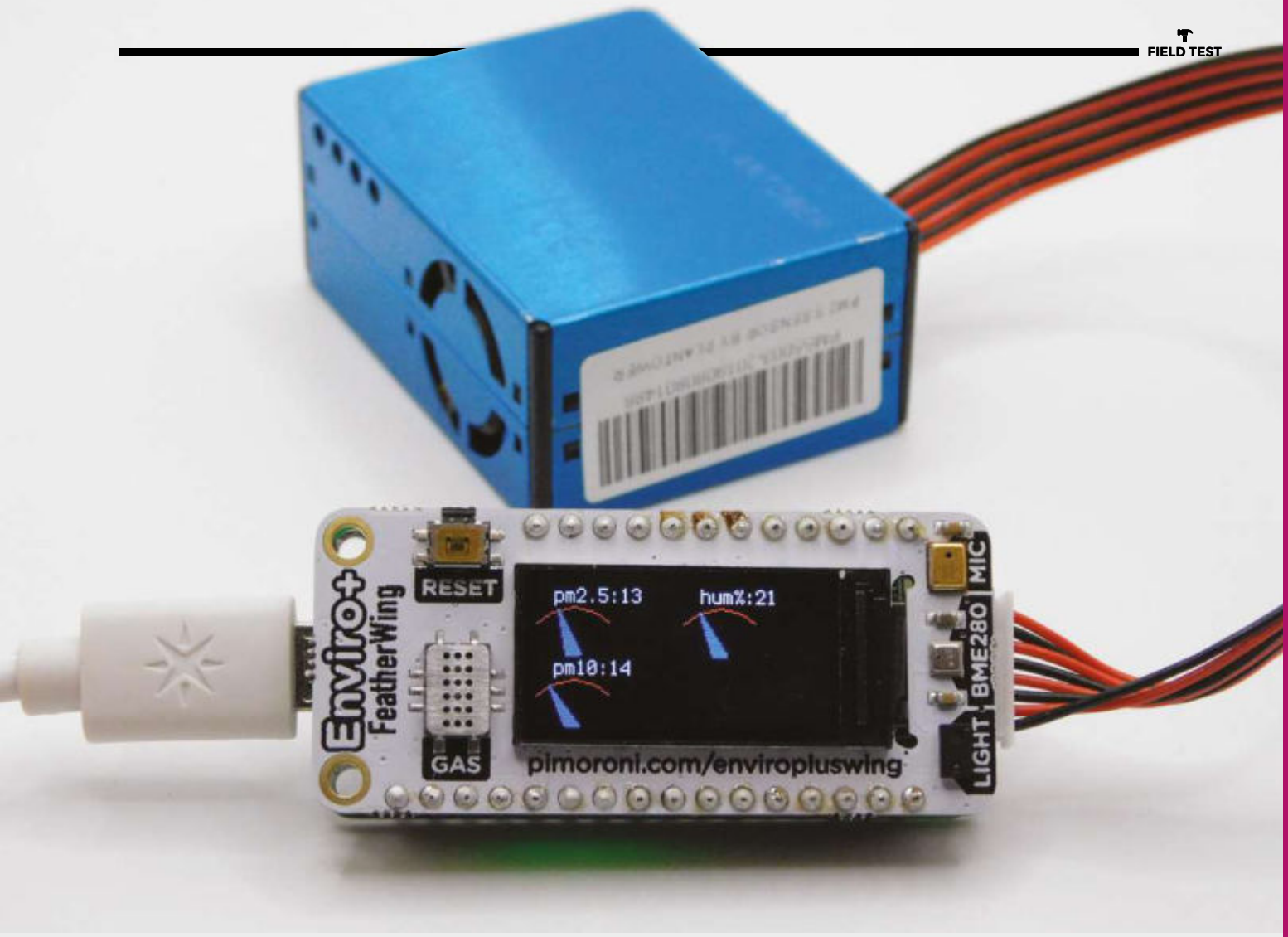
Perhaps the hardest sensor to understand is the gas sensor. It doesn't measure gases per se, but the properties of the gases in three areas: oxidising (which is typically nitrous dioxide), reducing (which is typically carbon monoxide), and ammonia. All are sensitive to other gasses, so you can't get an absolute value for the amount of any gas in the atmosphere, but you can use it to get a sense of whether the quality of the air is going up or down.

Of course, reading the sensors is only half the battle; you also need to do something with the data. For this, the Enviro+ FeatherWing comes with a 160×80 display which takes up the majority of the space on the Wing.

The Wing needs a Feather board to work, though not all are supported. Pimoroni recommends the Feather M4 Express or Feather nRF52840 Express. The M0 Express is also listed as working. If you want to get it working with a different Feather, take a look at the physical Feather pins library as you'll need to get this working on your board.

Heating

We found that the temperature on our board was quite distorted by the heat of the electronics. The effect of this will be different depending on the particular Feather you use, as they'll all have different thermal properties. On the nRF52840 Sense at least,



ours was reading about five degrees hotter than the ambient temperature, but changes to the code could probably decrease this.

Connected with a Feather, you can display any (or all) of the sensor values on the screen. The example code comes with a plotter that shows how the readings change over time. The screen is only 0.96 inches, but the resolution is high enough that you can display quite a bit of data.

The board itself has no connectivity, so if you want to read the data wirelessly, you'll need that either on the Feather you use (the nRF52840 has BLE), or as an additional FeatherWing. Since FeatherWings stack, you should be able to add WiFi or LoRa via the appropriate Wing should you need it. Similarly, no additional IOs are broken out, so if you want to add more sensors, you'll either need to carefully select which headers you connect, or break them out using a Wing multiplexer.

> We found that the temperature on our board **was quite distorted by the heat of the electronics**

TAKING FLIGHT

The Feather ecosystem is a great fit for a device like this. It's small, easy to use, and there's a great range of boards to expand it with the capabilities you need. Perhaps the most curious thing about the Feather ecosystem is why there hasn't been a good environmental sensor Wing before. The Enviro+ FeatherWing rights that wrong.

The example CircuitPython code shows you how to get data out of all the sensors with just a couple of lines of code – it all worked flawlessly for us. If you simply want to read data and show it on the screen, you probably won't need any more than the example code. However, if you want to send it somewhere (either via Bluetooth to a phone using an nRF52840 Feather or over some other network using additional Wings), you should find it easy to adapt the code to do what you need. ◻

Above
The display works well with our gauge library (see page 70)

VERDICT

Air quality should be important to all of us, and this helps us understand it.

9/10

REVIEW

Raspberry Pi High Quality Camera

Does the new Raspberry Pi camera live up to its name?

RASPBERRY PI (Trading) ◆ **$50 (lens sold separately)** | **raspberrypi.org**

By **Ben Everard** @ben_everard

Raspberry Pi has released a new High Quality Camera. For $50, you get a 12.3-megapixel sensor with 1.55µm × 1.55µm pixel size (that's double the pixel area of the standard Raspberry Pi Camera Module). It's also got a back-illuminated sensor architecture for improved sensitivity and support for off-the-shelf C- and CS-mount lenses. You can secure it to your project using either the mounting holes or the integrated tripod mount.



Below With the recommended 16mm, 10MP telephoto lens, you can shoot pictures at a distance

As well as this camera, you'll need a C or CS lens. There are two available from most resellers: a 6mm wide-angle 3MP lens for around £25, and a 16mm telephoto 10MP lens for around £50. The megapixel ratings of the lenses give you an idea of the image quality you can expect – you'll still be able to get the full 12.3MP out of the camera, but it won't have a high level of detail if you expand it to the full pixel range. You don't have to use one of these two, as any lens with C or CS mounting should work.

For makers, there are some things to really like about this Camera Module. Most obviously, it takes good pictures – much better than ones using any other commonly available hackable module we've found (see the inside back cover for some examples). As well as better pictures, you can take more of them. The HQ Camera Module is capable of up to 120fps video, which means it can capture fast-moving events. For us, though, the big thing about this new camera is its flexibility. The lens mounting system means you can use a staggering range of different lenses. There's already an ecosystem of C and CS lenses you can use, and a range of adaptors that can connect to other types of camera lens mounts. If you've already got a selection of camera lenses, it's worth checking if there's an adaptor that can link them to the HQ Camera Module.

DIFFERENT SETUPS

Because of the ability to use different lenses, this camera can behave quite differently depending on how you set it up.

For example, let's talk a little about focussing. The other Raspberry Pi Camera Module (like almost all maker-level camera modules) is fixed focus. This means that you can't change the focus – well, you can a little bit, but it's not very easy or accurate. However, the lens is set up so that a huge amount is in focus. Similarly, the computer can't change the zoom or aperture, so these are fixed to values that should work in the majority of situations. On the new High Quality Camera, the computer also can't change the focus, zoom, or aperture, but depending on the lens you add, there might be manual adjusters for these, and there may not be a general setting that's good for the majority of circumstances. This means that if you want a camera that you pop on and don't have to worry about adjusting, you need to pick a suitable lens. Some lenses will have a small depth of field and may need to be adjusted between shots, unless you have a very fixed setup.



Two things you will want to consider with the new camera are its size and cost. It's a much bigger and heavier beast than the other official Camera Module, so you'll need a beefier drone if you want to get this thing in the sky. At $50 (plus lenses), this isn't a cheap device but it is a powerful camera. While there are a few projects where you can simply drop this in place of a Raspberry Pi Camera Module and get better pictures (security cameras are one example), the real power of this is projects which wouldn't be possible with simpler camera modules. Do you want a wildlife camera where you can position the camera away from the place you want to photograph? Do you want to do detailed mapping from a drone or helium balloon? Do you want to mount the camera securely on a microscope or other specialist lens? Things like these were possible before, but only by connecting a full camera to a Raspberry Pi computer and controlling it over a USB cable. Compared to this setup, the High Quality Camera is much cheaper, lighter, and more configurable.

> "The High Quality Camera **is much cheaper, lighter, and more configurable**"

The High Quality Camera is definitely capable of taking high-quality photographs. However, it's not simply an upgrade to Raspberry Pi Camera Module, it's a unique product that opens up a whole new class of build. We're looking forward to seeing what you do with it. ◻



Above
With the recommended 6mm, 3MP lens, you get a wider field of view, but can still zoom in on the area you want

Left
The High Quality Camera needs a lens in order to be able to take pictures

VERDICT

A flexible camera that enables a new type of project.

10/10

REVIEW

Ultimate Box Maker

3D-print your own project enclosure

JBEBEL Free | Thingiverse.com

By **Ben Everard** @ben_everard



Above The four parts slot together easily

The final stage of a project is usually finding a box to house everything. There are loads available from online stores – or, if you have access to a laser cutter, some online services to help you design one. However, for many makers, a 3D printer is the go-to tool for digital fabrication, and that's where the Ultimate Box Maker comes in.

This is a parametric design in OpenSCAD, so you'll first need to download and install this bit of software. Once you've got this, you can grab the file from Thingiverse at **hsmag.cc/fLtADy**. This version is by user jbebel, and it's built upon the work of a user who goes by the name Heartman.

When you open it up in OpenSCAD, you should see the customizer panel on the right-hand side of the window, and this will let you enter the details you need for your box.



Below The four points for attaching PCBs help keep your circuits safe

The box is made of four parts: top and bottom shells that clip together and also make two sides of the box, and front and back panels that are flat pieces which slot into grooves. The whole thing is held together with four M2 screws (or there is an experimental screwless version). Inside the box, there is the option to add four mounting holes for PCBs.

The Ultimate Box Maker does some things really well. With a set of callipers (or a detailed specification), you can design a box to hold your PCB really quickly, and the front and back panels are easy to customise by adding raised text or holes (you have to edit the OpenSCAD code to do this, but this is quite straightforward).

The two main limitations are the restriction to exactly four PCB mounts (unless you modify the source code), which is limiting for more complex projects, and it's quite hard to include holes in the sides of the box rather than the front and back panels. For many projects, these two limits won't present a problem – but, for projects that do find these an issue, you can modify the STLs created in another CAD program, or modify the OpenSCAD code to get the box you need. This is an excellent resource, and just a few minutes entering sizes will leave you with a 3D-printable STL to keep your project safe when being knocked about.

VERDICT

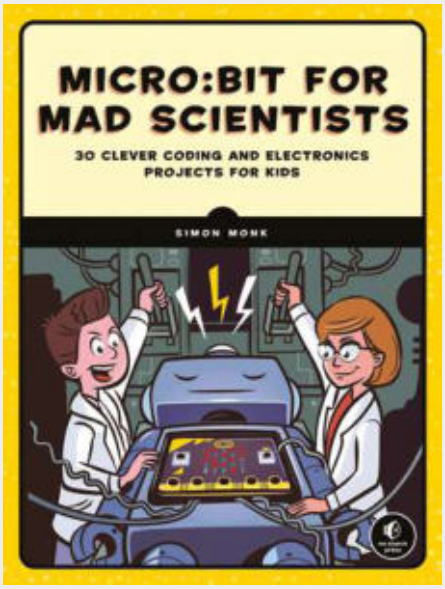
A great tool, but with some limitations for more complex projects.

8/10

Micro:bit for Mad Scientists

Simon Monk ◆ $24.95 | No Starch

By Andrew Gregory @AndrewGregory83



***Micro:bit for Mad Scientists* is aimed at kids aged 10+.** You get a thorough introduction to the micro:bit, including what it does, what sensors it includes, what you can add to it, how to add code to it, and everything else you might want to know when starting out with it.

Code examples are given in MicroPython, as well as Blocks, the graphical version that's very similar to Scratch, and all code is available to download.

One oddity of the first few pages is that the first example given is a Hello World program, which is perfectly traditional, but has to be the most boring way possible to introduce a technology with as much potential as the micro:bit. The other moment that jarred is that, moments after being told that the micro:bit is packed with sensors, the first hardware project requires an additional piece of hardware. This is a minor quibble – the whole point of physical computing is that you're adding other bits of hardware to a computer, whether that hardware is a microphone, motors, servos, speakers, or whatever. It's just a little odd that the capabilities inherent in the micro:bit aren't explored further before add-ons are introduced.

The rest of the book is packed with projects and experiments for young, mad scientists, incorporating servos, motors, music, communicating with Bluetooth Low Energy radio (BLE), and more. If you're a regular reader, you'll remember that we published a how-to on BLE in issue 30, which just goes to show this book's ambition. It's aimed at kids, but there's nothing dumbed down here. *Micro:bit for Mad Scientists* is an excellent resource to take curious kids on a journey into physical computing. □

VERDICT

Thorough, fun, useful, and varied projects for the Emmett Brown in your life.

8/10

next month

issue #32

ON SALE 18 JUNE

EXTREME MAKES

PROJECTS THAT PUSH THE ENVELOPE

ALSO

→ ROBOTIC MUSIC

→ FIBREGLASS

→ PHOTOGRAMMETRY

→ 3D PRINTING

→ AND MUCH MORE

DON'T MISS OUT

hsmag.cc/subscribe







#ShotOnRaspberryPi

The new High Quality Camera lets you change lenses. This gives you the ability to set it up for a huge range of shots. Close-ups, wide angles, and long distance can all work, provided you strap the right bit of curved glass to the front. This gives us far more creative control than we can get on camera modules with fixed lenses.

There's a huge ecosystem of compatible lenses around at the moment thanks to the C and CS lens mounting, so you'll find ones capable of a wide range of shots, including all of the above.

